BAB 3. METODOLOGI PENELITIAN

3.1 Pendekatan Metodologi

Penelitian ini menggunakan pendekatan Research and Development (R&D) karena bertujuan untuk mengembangkan sebuah sistem yang dapat membantu pelaku usaha dalam memperoleh dan mengolah informasi produk dari platform 1688.com. Pendekatan ini dipilih karena tidak hanya berfokus pada analisis permasalahan, tetapi juga menghasilkan sebuah produk berupa sistem yang dapat digunakan untuk menyelesaikan permasalahan tersebut.

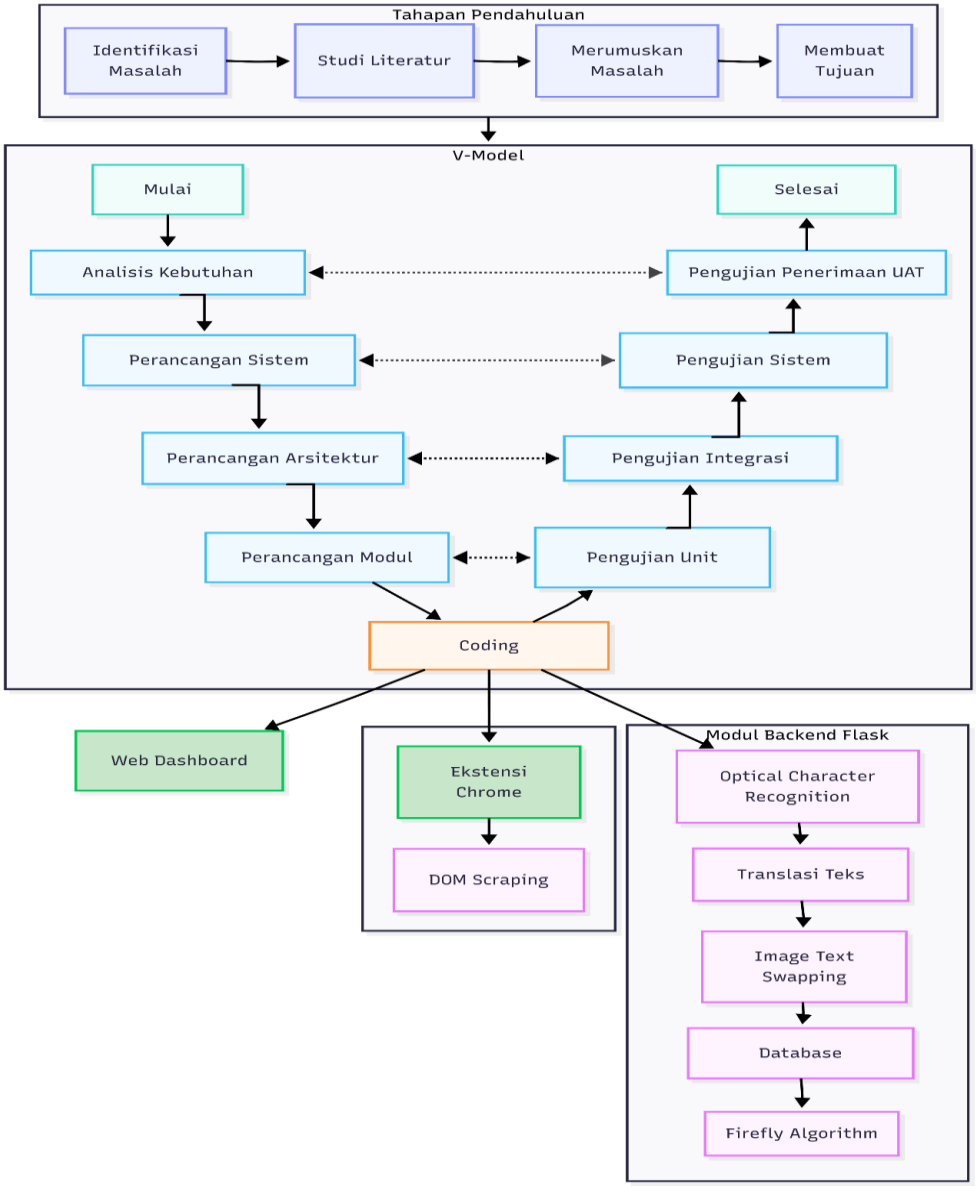
Model pengembangan sistem yang digunakan dalam penelitian ini adalah V-Model. Model ini dipilih karena memiliki tahapan pengembangan yang terstruktur, dimulai dari identifikasi kebutuhan, perancangan sistem, implementasi, hingga pengujian. Selain itu, setiap tahapan pengembangan memiliki proses verifikasi dan validasi sehingga sistem yang dikembangkan dapat dievaluasi secara sistematis untuk memastikan kesesuaiannya dengan kebutuhan pengguna.

3.2 Tahapan Penelitian

Penelitian ini menggunakan model pengembangan sistem V-Model sebagai acuan dalam proses pengembangan sistem. V-Model dipilih karena menyediakan tahapan pengembangan yang sistematis serta menghubungkan setiap proses perancangan dengan tahapan pengujian yang sesuai. Melalui pendekatan ini, setiap hasil yang diperoleh pada tahap pengembangan dapat diverifikasi dan divalidasi secara bertahap sehingga sistem yang dihasilkan sesuai dengan kebutuhan pengguna.

Sebelum memasuki proses pengembangan menggunakan V-Model, penelitian diawali dengan tahap pendahuluan yang terdiri atas identifikasi masalah, studi literatur, perumusan masalah, dan penetapan tujuan penelitian. Tahap ini bertujuan untuk memahami permasalahan penelitian, menentukan ruang lingkup penelitian, serta memperoleh landasan teori dan metode yang relevan sebagai dasar dalam pengembangan sistem.

Selanjutnya proses pengembangan sistem dilakukan menggunakan metode V-Model yang terdiri atas tahapan analisis kebutuhan sistem, perancangan sistem, perancangan arsitektur, perancangan modul, implementasi (*coding*), serta pengujian yang meliputi Unit Testing, Integration Testing, System Testing, dan User Acceptance Testing (UAT). Hubungan antara tahapan penelitian dan proses pengembangan sistem menggunakan V-Model ditunjukkan pada Gambar 3.1.



Gambar 3.1 Tahapan Penelitian Menggunakan V-Model

Berdasarkan Gambar 3.1, penelitian ini terdiri atas dua bagian utama, yaitu tahap pendahuluan dan tahap pengembangan sistem menggunakan V-Model. Tahap pendahuluan menghasilkan dasar penelitian berupa identifikasi permasalahan, rumusan masalah, tujuan penelitian, serta kajian literatur yang selanjutnya digunakan sebagai acuan pada proses pengembangan sistem.

Tahap pertama pada V-Model adalah analisis kebutuhan sistem, yaitu mengidentifikasi kebutuhan fungsional dan nonfungsional yang harus dipenuhi oleh sistem berdasarkan hasil identifikasi masalah dan studi literatur. Hasil dari tahap ini menjadi dasar dalam penyusunan rancangan sistem.

Tahap berikutnya adalah perancangan sistem, yaitu menyusun rancangan solusi berdasarkan kebutuhan yang telah diperoleh. Pada tahap ini dilakukan penyusunan gambaran umum sistem, proses bisnis, serta kebutuhan fungsional yang akan diimplementasikan.

Selanjutnya dilakukan perancangan arsitektur (*Architecture Design*) untuk menentukan struktur sistem beserta hubungan antar komponen utama. Hasil dari tahap ini berupa rancangan arsitektur yang terdiri atas Chrome Extension, Backend Flask, dan Web Dashboard sebagai komponen utama sistem.

Setelah rancangan arsitektur selesai disusun, dilakukan perancangan modul (*Module Design*) dengan mendefinisikan fungsi setiap modul yang akan dikembangkan. Modul yang dirancang meliputi DOM Scraping pada Chrome Extension, *Optical Character Recognition (OCR)*, Google Translate API, *Image Text Swapping*, *Database*, serta *Firefly Algorithm* pada *Backend Flask*.

Tahap implementasi (*Coding*) merupakan proses pembangunan sistem berdasarkan seluruh hasil perancangan yang telah disusun. Implementasi menghasilkan tiga komponen utama, yaitu Web Dashboard sebagai media pengolahan data hasil scraping, Chrome Extension sebagai antarmuka pengguna pada platform 1688.com yang menjalankan proses DOM Scraping, serta Modul Backend Flask yang mengimplementasikan proses Optical Character Recognition (OCR), penerjemahan teks, Image Text Swapping, pengelolaan database, dan Firefly Algorithm sebagai mekanisme optimasi pada sistem.

Setelah proses implementasi selesai, sistem memasuki tahap pengujian dan validasi yang terdiri atas *Unit Testing*, *Integration Testing*, *System Testing*, dan *User Acceptance Testing (UAT)*. Pengujian dilakukan secara bertahap untuk memastikan bahwa setiap modul, integrasi antar komponen, serta keseluruhan fungsi sistem telah berjalan sesuai dengan kebutuhan pengguna dan tujuan penelitian.

Dengan demikian, setiap tahapan pengembangan pada sisi kiri V-Model memiliki keterkaitan langsung dengan tahapan pengujian pada sisi kanan. Pendekatan ini memastikan bahwa setiap hasil perancangan telah melalui proses verifikasi dan validasi sebelum sistem digunakan. Hubungan antara tahapan pengembangan dan tahapan pengujian pada V-Model dirangkum pada Tabel 3.1.

Tabel 3.1 Hubungan Tahapan Pengembangan dan Tahapan Pengujian pada V-Model

Fase Perancangan (Kiri)	Hubungan Pengujian (Kanan)	Penjelasan Hubungan
Analisis Kebutuhan Sistem	Pengujian Penerimaan (UAT)	Pengujian dilakukan berdasarkan kebutuhan pengguna untuk memastikan sistem memenuhi ekspektasi dan tujuan awal.
Perancangan Sistem	Pengujian Sistem	Menguji apakah sistem secara keseluruhan bekerja sesuai rancangan fungsional dan non-fungsional yang telah dibuat.
Arsitektur Design	Pengujian Integrasi	Memastikan modul-modul yang dirancang dapat berkomunikasi dan terintegrasi tanpa error.
Module Design	Pengujian Unit	Menguji setiap modul atau komponen kecil (seperti scraping, OCR, translasi) agar berfungsi sesuai spesifikasi sebelum digabungkan.
Coding (Implementasi)	—	Tahap implementasi utama yang menjadi jembatan antara perancangan dan pengujian seluruh sistem.

3.2.1 Identifikasi Masalah dan Studi Literatur

Tahap pertama dalam penelitian ini adalah melakukan identifikasi masalah dan studi literatur. Identifikasi masalah bertujuan untuk memperoleh pemahaman mengenai kondisi yang dihadapi pelaku usaha dalam memperoleh informasi produk dari platform 1688.com serta mengidentifikasi kendala yang muncul selama proses tersebut.

Identifikasi masalah dilakukan melalui observasi terhadap platform 1688.com dan proses bisnis pengadaan produk yang dilakukan oleh pelaku usaha. Hasil observasi menunjukkan bahwa sebagian besar informasi penting mengenai spesifikasi, keunggulan, serta deskripsi produk disajikan dalam bentuk gambar yang mengandung teks berbahasa Mandarin. Kondisi tersebut menyebabkan informasi tidak dapat diterjemahkan secara langsung menggunakan penerjemah

berbasis halaman web, sehingga proses analisis produk menjadi kurang efektif dan memerlukan waktu yang lebih lama.

Selanjutnya dilakukan studi literatur dengan mempelajari berbagai referensi ilmiah yang berkaitan dengan *web scraping*, *Optical Character Recognition* (OCR), Google Translate API, *image text swapping*, Large Language Model (LLM), Chrome Extension, serta Firefly Algorithm. Studi literatur dilakukan untuk memperoleh landasan teoritis, memahami perkembangan penelitian terdahulu, serta menentukan metode dan teknologi yang sesuai dengan kebutuhan penelitian.

Hasil dari tahap ini berupa rumusan permasalahan, tujuan penelitian, serta dasar pemilihan metode dan teknologi yang digunakan dalam pengembangan sistem.

3.2.2 Analisis Kebutuhan Sistem

Setelah permasalahan penelitian berhasil diidentifikasi, tahap selanjutnya adalah melakukan analisis kebutuhan sistem. Tahap ini bertujuan untuk mengidentifikasi kebutuhan yang harus dipenuhi oleh sistem agar mampu menyelesaikan permasalahan yang ditemukan selama proses observasi.

Analisis kebutuhan dilakukan berdasarkan hasil observasi terhadap platform 1688.com, studi literatur, serta identifikasi proses bisnis yang dilakukan oleh pelaku usaha dalam memperoleh informasi produk impor. Pada tahap ini dilakukan identifikasi terhadap kebutuhan fungsional dan nonfungsional sistem, termasuk kebutuhan terkait proses pengambilan data produk (*web scraping*), ekstraksi teks pada gambar menggunakan *Optical Character Recognition* (OCR), penerjemahan informasi produk, pengelolaan data hasil scraping, serta penyajian informasi melalui dashboard.

3.2.3 Perancangan Sistem

Tahap perancangan sistem dilakukan setelah kebutuhan sistem berhasil diidentifikasi. Tujuan dari tahap ini adalah menyusun rancangan sebagai acuan dalam proses pengembangan sistem sehingga implementasi dapat dilakukan secara terstruktur dan sesuai dengan kebutuhan yang telah ditetapkan.

Pada tahap ini dilakukan perancangan terhadap arsitektur sistem, alur proses, basis data, antarmuka pengguna, serta mekanisme integrasi antar komponen sistem. Perancangan juga mencakup penyusunan alur proses *web scraping*, ekstraksi teks pada gambar menggunakan *Optical Character Recognition* (OCR), proses penerjemahan informasi produk, serta pengelolaan data hasil scraping yang akan disajikan melalui dashboard.

3.2.4 Implementasi Sistem

Tahap implementasi sistem merupakan proses penerapan hasil perancangan ke dalam bentuk aplikasi yang dapat dijalankan. Pada tahap ini seluruh rancangan yang telah disusun pada tahap sebelumnya direalisasikan menjadi sebuah sistem yang berfungsi sesuai dengan kebutuhan yang telah ditetapkan.

Implementasi dilakukan dengan mengembangkan setiap komponen sistem secara bertahap, meliputi ekstensi Google Chrome sebagai media interaksi pengguna, aplikasi web sebagai dashboard pengelolaan data, serta layanan backend yang bertugas mengelola proses pengambilan data produk, pengolahan informasi, dan penyimpanan data. Seluruh komponen tersebut diintegrasikan sehingga mampu bekerja sebagai satu kesatuan sistem.

3.2.5 Pengujian dan Validasi

Tahap terakhir dalam penelitian ini adalah pengujian dan validasi sistem. Tahap ini bertujuan untuk memastikan bahwa sistem yang telah diimplementasikan dapat berfungsi sesuai dengan kebutuhan yang telah ditetapkan pada tahap analisis serta memenuhi tujuan penelitian.

Pengujian dilakukan secara bertahap mengikuti pendekatan V-Model, yang meliputi *Unit Testing*, *Integration Testing*, *System Testing*, dan *User Acceptance Testing* (UAT). *Unit Testing* dilakukan untuk memastikan setiap modul sistem dapat berfungsi dengan baik secara mandiri. *Integration Testing* bertujuan untuk memverifikasi bahwa seluruh modul dapat saling terintegrasi dan berkomunikasi dengan baik. Selanjutnya, *System Testing* dilakukan untuk menguji keseluruhan fungsi sistem sesuai dengan kebutuhan yang telah dirancang. Tahap terakhir adalah *User Acceptance Testing* (UAT) yang dilakukan untuk mengevaluasi apakah sistem telah memenuhi kebutuhan pengguna dan dapat digunakan sesuai dengan tujuan pengembangannya. Melalui proses pengujian dan validasi tersebut, sistem diharapkan mampu bekerja secara optimal dalam mendukung proses pengambilan informasi produk dari platform 1688.com.

3.2.6 Teknik Pengumpulan Data

Teknik pengumpulan data merupakan tahapan yang dilakukan untuk memperoleh informasi yang dibutuhkan sebagai dasar dalam analisis kebutuhan sistem. Data yang dikumpulkan pada penelitian ini berasal dari berbagai sumber, baik berupa referensi ilmiah maupun hasil observasi langsung terhadap platform 1688.com. Teknik pengumpulan data yang digunakan meliputi studi literatur, observasi, dan dokumentasi.

3.2.7 Studi Literatur

Studi literatur dilakukan dengan mempelajari berbagai sumber ilmiah, seperti buku, jurnal, prosiding, dokumentasi resmi, dan penelitian terdahulu yang berkaitan dengan topik penelitian. Literatur yang dikaji mencakup teknologi *web scraping*, Optical Character Recognition (OCR), Google Translate API, image text swapping, Large Language Model (LLM), Chrome Extension, Firefly Algorithm, serta teknologi pendukung lainnya yang relevan dengan pengembangan sistem.

Hasil studi literatur digunakan sebagai landasan teoritis dalam penelitian, membantu pemilihan metode yang sesuai, serta menjadi acuan dalam proses pengembangan dan evaluasi sistem yang dibangun.

3.2.8 Observasi

Observasi dilakukan secara langsung terhadap platform 1688.com untuk memahami proses pencarian dan pengelolaan informasi produk oleh pelaku usaha impor. Kegiatan observasi meliputi pengamatan terhadap alur pencarian produk, struktur halaman produk, mekanisme pemuatan data pada halaman web, serta informasi produk yang tersedia dalam bentuk teks maupun gambar.

Selain mengamati karakteristik platform, observasi juga dilakukan terhadap proses penyusunan katalog produk, mulai dari pencarian produk, penerjemahan informasi, hingga penyusunan data katalog. Hasil observasi tersebut digunakan sebagai dasar dalam mengidentifikasi permasalahan pada sistem berjalan serta merumuskan kebutuhan sistem yang akan dikembangkan.

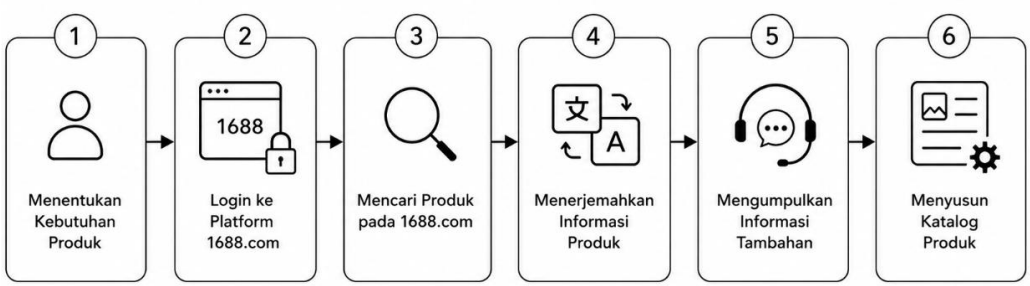
3.3 Analisis Sistem

Analisis sistem merupakan tahapan yang dilakukan untuk memahami proses bisnis yang berjalan, mengidentifikasi permasalahan, serta merumuskan kebutuhan sistem yang akan dikembangkan. Tahap ini dilakukan berdasarkan hasil observasi terhadap platform 1688.com serta proses pencarian, pengumpulan, dan pengelolaan informasi produk yang dilakukan oleh pelaku usaha impor. Selain itu, analisis juga mempertimbangkan kebutuhan pengguna terhadap proses *web scraping*, penerjemahan informasi produk, dan *image text swapping*. Hasil analisis tersebut menjadi dasar dalam penyusunan rancangan sistem yang dibahas pada Bab IV.

3.3.1 Analisis Proses Bisnis

Analisis proses bisnis dilakukan untuk memahami alur kerja yang saat ini diterapkan oleh pelaku usaha dalam memperoleh informasi produk dari platform 1688.com hingga menyusun katalog produk. Analisis ini bertujuan untuk

mengidentifikasi aktivitas yang masih dilakukan secara manual sehingga dapat diketahui peluang penerapan sistem dalam meningkatkan efisiensi proses kerja.



Gambar 3.2 Alur Proses Bisnis Pengadaan Data Produk dari Platform 1688.com

Berdasarkan Gambar 3.2, proses bisnis diawali dengan penentuan kebutuhan produk oleh pelaku usaha, kemudian dilanjutkan dengan proses login ke platform 1688.com, pencarian produk, penerjemahan informasi produk, pengumpulan informasi tambahan dari supplier, hingga penyusunan katalog produk. Seluruh tahapan tersebut masih dilakukan secara manual dengan menggunakan beberapa aplikasi atau layanan yang berbeda sehingga proses pengumpulan dan pengolahan informasi produk memerlukan waktu yang relatif lama serta berpotensi menimbulkan kesalahan.

Berdasarkan analisis proses bisnis tersebut, diketahui bahwa proses pengolahan informasi produk masih melibatkan berbagai aktivitas manual, seperti pengambilan data produk, penerjemahan teks pada gambar, pengumpulan informasi tambahan, serta penyusunan katalog produk. Selain itu, penggunaan beberapa aplikasi atau layanan secara terpisah menyebabkan alur kerja menjadi kurang efisien serta meningkatkan potensi kesalahan dalam pengolahan data. Oleh karena itu, diperlukan suatu sistem yang mampu mengintegrasikan proses DOM Scraping, *Optical Character Recognition (OCR)*, penerjemahan teks, *Image Text Swapping*, serta pengelolaan data produk ke dalam satu alur kerja yang terintegrasi sehingga proses penyusunan katalog produk dapat dilakukan secara lebih efisien. Hasil analisis tersebut menjadi dasar dalam proses analisis kebutuhan sistem pada sub bab berikutnya.

3.3.2 Identifikasi Permasalahan

Berdasarkan hasil analisis proses bisnis yang telah dijelaskan pada Sub Bab 3.4.1, diketahui bahwa proses pengumpulan informasi produk dari platform 1688.com masih menghadapi beberapa kendala yang menyebabkan proses kerja menjadi kurang efisien. Kendala tersebut berkaitan dengan proses pengumpulan informasi, penerjemahan teks, pengolahan gambar produk, serta penyusunan katalog yang masih dilakukan secara manual. Oleh karena itu, identifikasi

permasalahan dilakukan sebagai dasar dalam merumuskan kebutuhan sistem yang akan dikembangkan.

Ringkasan hasil identifikasi permasalahan pada sistem berjalan disajikan pada Tabel 3.2.

Tabel 3.2 Analisis Permasalahan Sistem Berjalan

No	Permasalahan	Dampak
1	Informasi produk menggunakan bahasa Mandarin.	Pelaku usaha mengalami kesulitan dalam memahami spesifikasi dan informasi produk secara cepat.
2	Teks pada gambar produk belum dapat diterjemahkan dan diganti secara otomatis.	Pengguna harus melakukan OCR, menerjemahkan, dan mengganti teks pada gambar secara manual sehingga proses menjadi kurang efisien.
3	Belum terdapat sistem yang mengintegrasikan proses *web scraping*, OCR, penerjemahan, *image text swapping*, dan penyusunan informasi produk.	Pengguna harus berpindah antar aplikasi sehingga proses kerja menjadi kurang efisien dan meningkatkan risiko kesalahan dalam pengolahan informasi.
4	Penyusunan katalog produk masih dilakukan secara manual dalam pengubahan bahasa.	Membutuhkan waktu dan tenaga yang lebih banyak dalam menyusun katalog produk.

Berdasarkan hasil identifikasi permasalahan pada Tabel 3.2, dapat disimpulkan bahwa proses pengelolaan informasi produk pada platform 1688.com masih menghadapi beberapa kendala, seperti perbedaan bahasa, keterbatasan penerjemahan teks pada gambar, serta penggunaan beberapa aplikasi secara terpisah dalam proses *web scraping*, OCR, penerjemahan, dan penyusunan katalog. Kondisi tersebut menyebabkan proses kerja menjadi kurang efisien serta meningkatkan potensi kesalahan dalam pengolahan informasi. Oleh karena itu, diperlukan suatu sistem yang mampu mengintegrasikan seluruh proses tersebut dalam satu platform. Analisis kebutuhan sistem untuk memenuhi kebutuhan tersebut dibahas pada sub bab berikutnya.

3.3.3 Kebutuhan Sistem

Analisis kebutuhan sistem dilakukan untuk merumuskan kebutuhan yang harus dipenuhi oleh sistem berdasarkan hasil analisis proses bisnis, identifikasi permasalahan, dan observasi terhadap platform 1688.com. Tahap ini bertujuan untuk memastikan bahwa sistem yang dikembangkan mampu membantu pelaku usaha dalam proses pengadaan data produk secara lebih efektif dan efisien.

Kebutuhan sistem pada penelitian ini diperoleh melalui observasi terhadap proses pengumpulan informasi produk, karakteristik platform 1688.com, serta kendala yang dihadapi oleh pelaku usaha selama proses pencarian, penerjemahan, dan penyusunan informasi produk. Berdasarkan hasil observasi dan analisis tersebut, kebutuhan sistem dikelompokkan menjadi kebutuhan fungsional dan kebutuhan non fungsional.

3.3.3.1 Analisis Kebutuhan Berdasarkan Observasi

Observasi dilakukan secara langsung terhadap platform 1688.com dengan mengamati alur penggunaan, struktur halaman produk, serta proses pengumpulan informasi produk yang dilakukan oleh pelaku usaha. Observasi juga dilakukan terhadap aktivitas penyusunan katalog produk untuk mengidentifikasi kebutuhan sistem yang sesuai dengan kondisi nyata pada proses bisnis.

Hasil observasi menunjukkan bahwa proses pengumpulan informasi produk masih menghadapi beberapa kendala, antara lain penggunaan bahasa Mandarin pada informasi produk, keberadaan informasi penting dalam bentuk gambar (*in-image text*), penggunaan beberapa aplikasi secara terpisah, serta penyusunan katalog produk yang masih dilakukan secara manual. Selain itu, halaman produk pada platform 1688.com memuat informasi secara dinamis sehingga diperlukan mekanisme yang mampu memperoleh data secara lengkap dari halaman yang telah dimuat pada browser.

Ringkasan hasil observasi dan kebutuhan sistem yang diperoleh disajikan pada Tabel 3.3.

Tabel 3.3 Kebutuhan Sistem

No	Hasil Observasi	Kebutuhan Sistem
1	Informasi produk masih menggunakan bahasa Mandarin.	Sistem mampu menerjemahkan informasi produk ke dalam bahasa Indonesia

No	Hasil Observasi	Kebutuhan Sistem
		sehingga lebih mudah dipahami oleh pelaku usaha.
2	Sebagian besar informasi penting berada pada gambar produk (*in-image text*).	Sistem mampu mendeteksi teks pada gambar, menerjemahkannya, serta menampilkan kembali hasil terjemahan pada gambar produk (*image text swapping*).
3	Halaman produk dimuat secara dinamis (*dynamic rendering*).	Sistem mampu melakukan *DOM scraping* pada halaman yang telah dimuat sehingga informasi produk dapat diperoleh secara lengkap.
4	Pengaturan parameter setiap tahapan pemrosesan memengaruhi efisiensi pipeline sistem.	Sistem mampu mengoptimalkan konfigurasi *pipeline* menggunakan Firefly Algorithm.
5	Pengumpulan informasi produk masih melibatkan beberapa aplikasi yang berbeda.	Sistem mampu mengintegrasikan proses *web scraping*, OCR, penerjemahan, *image text swapping*, penyusunan deskripsi produk, dan penyimpanan data dalam satu sistem.
6	Penyusunan katalog produk masih dilakukan secara manual.	Sistem mampu menyimpan hasil pengolahan informasi produk ke dalam basis data serta menyajikannya melalui *Web Dashboard* untuk mempermudah penyusunan katalog produk.

Berdasarkan hasil observasi tersebut, diperoleh kebutuhan sistem yang menjadi dasar dalam penyusunan kebutuhan fungsional dan kebutuhan non fungsional pada penelitian ini.

3.3.3.2 Kebutuhan Fungsional

Kebutuhan fungsional merupakan fungsi utama yang harus dimiliki oleh sistem agar dapat mendukung proses pengumpulan dan pengolahan informasi

produk sesuai dengan kebutuhan pengguna. Kebutuhan ini disusun berdasarkan hasil observasi dan identifikasi permasalahan pada sistem yang berjalan.

Tabel 3.4 Kebutuhan Fungsional Sistem

Kode	Kebutuhan Fungsional
F-01	Sistem mampu mengambil gambar produk dari halaman 1688.com secara otomatis.
F-02	Sistem mampu memperoleh informasi produk melalui *DOM scraping* pada halaman yang telah dimuat di browser.
F-03	Sistem mampu mendeteksi teks pada gambar produk menggunakan OCR.
F-04	Sistem mampu menerjemahkan hasil ekstraksi teks menggunakan Google Translate API.
F-05	Sistem mampu menampilkan hasil terjemahan kembali pada gambar produk (*image text swapping*).
F-06	Sistem mampu menghasilkan nama produk, deskripsi produk, dan kata kunci menggunakan Gemini.
F-07	Sistem mampu mengoptimalkan konfigurasi *Pipeline Scheduler* menggunakan Firefly Algorithm.
F-08	Sistem mampu menyimpan hasil pengolahan informasi produk ke dalam basis data.
F-09	Sistem mampu menampilkan informasi produk melalui Web Dashboard.
F-10	Sistem mampu melakukan pencarian dan penyaringan informasi produk.
F-11	Sistem mampu mengunduh hasil gambar terjemahan dan informasi produk.

Berdasarkan kebutuhan fungsional tersebut, sistem diharapkan mampu mengotomatisasi proses pengumpulan dan pengolahan informasi produk mulai dari proses *DOM scraping*, OCR, penerjemahan, *image text swapping*, optimasi *Pipeline Scheduler* menggunakan Firefly Algorithm, hingga penyajian informasi produk melalui Web Dashboard dalam satu sistem yang terintegrasi.

3.3.3.3 Kebutuhan Non-Fungsional

Selain kebutuhan fungsional, sistem juga memiliki kebutuhan non-fungsional yang berkaitan dengan kualitas sistem. Kebutuhan non-fungsional digunakan untuk memastikan sistem memiliki karakteristik operasional yang baik sehingga mampu berjalan secara andal, mudah digunakan, kompatibel dengan lingkungan implementasi, serta menjaga konsistensi data selama proses pengolahan informasi produk.

Tabel 3.5 Kebutuhan Non-Fungsional Sistem

Kode	Kebutuhan Non-Fungsional
NF-01	Sistem dapat dijalankan pada browser Google Chrome.
NF-02	Sistem memiliki antarmuka yang mudah digunakan (*user friendly*).
NF-03	Sistem mampu menjaga konsistensi dan integritas data selama proses pengolahan informasi produk berlangsung.
NF-04	Sistem mampu terhubung dengan layanan Google Translate API dan Gemini API selama proses penerjemahan dan penyusunan deskripsi produk.
NF-05	Sistem memerlukan koneksi internet selama proses *scraping*, penerjemahan, dan pengolahan data berlangsung.

3.4 Perancangan Sistem

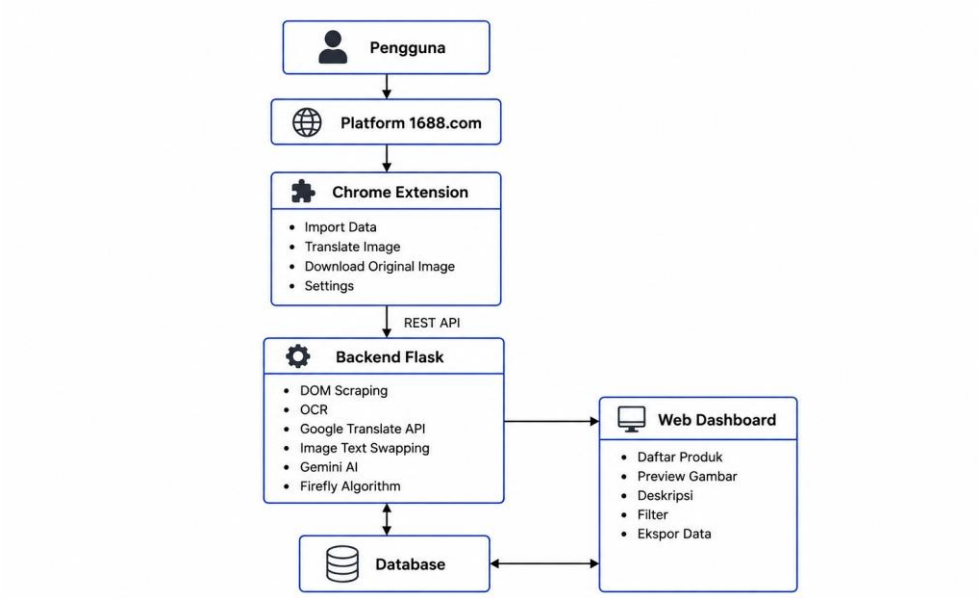
Berdasarkan hasil analisis sistem yang telah dilakukan, tahap selanjutnya adalah menyusun perancangan sistem sebagai acuan dalam proses implementasi. Perancangan sistem bertujuan untuk menghasilkan desain yang mampu memenuhi kebutuhan fungsional dan non-fungsional yang telah dirumuskan, sehingga sistem dapat mendukung proses pengadaan data produk dari platform 1688.com secara lebih efektif dan terintegrasi.

Perancangan sistem pada penelitian ini meliputi gambaran umum sistem, arsitektur sistem, pemodelan menggunakan *Unified Modeling Language* (UML), perancangan basis data, perancangan antarmuka pengguna, serta perancangan

modul sistem. Seluruh hasil perancangan tersebut menjadi dasar dalam proses implementasi sistem.

3.4.1 Gambaran Umum Sistem

Sistem yang diusulkan merupakan sistem pengumpulan dan pengolahan informasi produk berbasis Chrome Extension yang terintegrasi dengan Backend Flask, Database, dan Web Dashboard. Sistem ini dirancang untuk membantu pelaku usaha memperoleh informasi produk dari platform 1688.com secara lebih cepat dan efisien melalui proses pengambilan data produk, pengolahan informasi, serta pengelolaan hasil pemrosesan dalam satu sistem yang terintegrasi.



Gambar 3.3 Gambaran Umum Sistem

Berdasarkan Gambar 3.3, proses sistem diawali ketika pengguna mengakses halaman produk pada platform 1688.com menggunakan peramban Google Chrome yang telah dipasang Chrome Extension. Chrome Extension berfungsi sebagai antarmuka utama yang menyediakan fitur Import Data, Translate Image, Download Original Image, Settings, dan akses menuju Web Dashboard.

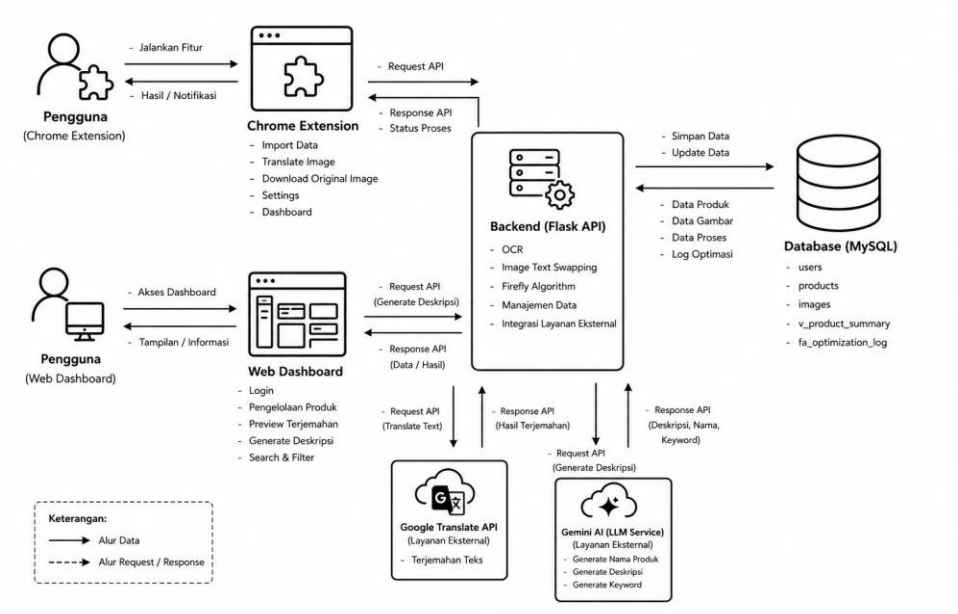
Setiap permintaan yang dilakukan melalui Chrome Extension dikirimkan ke Backend Flask melalui REST API untuk diproses. Backend Flask mengimplementasikan berbagai modul utama, yaitu DOM Scraping, Optical Character Recognition (OCR), Google Translate API, Image Text Swapping, Gemini AI, dan Firefly Algorithm. Seluruh hasil pemrosesan kemudian disimpan ke dalam Database sebagai media penyimpanan data produk dan hasil pengolahan.

Selanjutnya, data yang telah tersimpan pada Database dapat diakses melalui Web Dashboard untuk mendukung pengelolaan informasi produk. Dashboard menyediakan berbagai fungsi, seperti menampilkan daftar produk, meninjau hasil terjemahan gambar, melihat informasi produk yang dihasilkan sistem, melakukan pencarian dan penyaringan data, serta mengekspor data sesuai kebutuhan pengguna.

Secara umum, sistem yang diusulkan mengintegrasikan Chrome Extension, Backend Flask, Database, dan Web Dashboard ke dalam satu arsitektur yang saling terhubung. Integrasi tersebut memungkinkan proses pengambilan data, pengolahan informasi produk, penyimpanan data, serta pengelolaan hasil pemrosesan dilakukan secara lebih efisien dibandingkan proses manual.

3.5 Arsitektur Sistem Usulan

Arsitektur sistem usulan menggambarkan hubungan antar komponen yang membentuk sistem pengumpulan dan pengolahan informasi produk berbasis Chrome Extension pada platform 1688.com. Arsitektur ini menggunakan pendekatan client-server, di mana Chrome Extension berperan sebagai *client* yang berinteraksi langsung dengan halaman produk, sedangkan Backend Flask berfungsi sebagai pusat pemrosesan dan integrasi layanan sistem. Backend mengelola proses OCR, penerjemahan menggunakan Google Translate API, *Image Text Swapping*, integrasi dengan Gemini AI, serta optimasi pipeline menggunakan Firefly Algorithm. Seluruh hasil pemrosesan disimpan pada basis data MySQL dan dikelola melalui Web Dashboard. Arsitektur sistem usulan ditunjukkan pada Gambar 3.4.



Gambar 3.4 Arsitektur Sistem Usulan

Berdasarkan Gambar 3.4, proses sistem diawali ketika pengguna membuka halaman produk pada platform 1688.com menggunakan Google Chrome yang telah dipasang Chrome Extension. Melalui ekstensi tersebut, pengguna dapat menjalankan fitur Import Data, Translate Image, Download Original Image, Settings, serta mengakses Web Dashboard. Setiap permintaan dari Chrome Extension dikirimkan ke Backend Flask melalui REST API untuk diproses lebih lanjut.

Backend Flask berperan sebagai pusat pemrosesan sistem. Data yang diterima dari Chrome Extension diproses melalui beberapa tahapan, yaitu ekstraksi teks menggunakan Optical Character Recognition (OCR), penerjemahan menggunakan Google Translate API, serta penggantian teks pada gambar melalui mekanisme Image Text Swapping. Selain itu, Firefly Algorithm digunakan untuk mengoptimalkan konfigurasi pipeline pemrosesan sehingga proses pengolahan data dapat berlangsung secara lebih efisien.

Pada Web Dashboard, pengguna dapat mengelola data produk, melihat hasil terjemahan gambar, melakukan pencarian data, serta menghasilkan deskripsi melalui fitur Generate Deskripsi. Ketika fitur tersebut dijalankan, Web Dashboard mengirimkan permintaan ke Backend Flask, kemudian Backend berkomunikasi dengan Gemini AI untuk menghasilkan informasi produk berdasarkan data yang telah tersimpan pada basis data. Hasil yang diperoleh selanjutnya disimpan kembali ke dalam basis data dan ditampilkan pada Web Dashboard.

Seluruh data sistem disimpan pada Database MySQL yang terdiri atas tabel users, products, dan images, serta objek v_product_summary dan fa_optimization_log. Basis data digunakan untuk menyimpan informasi pengguna, data produk, hasil pengolahan gambar, ringkasan informasi produk, serta log proses optimasi Firefly Algorithm sehingga seluruh komponen sistem dapat mengakses data secara terpusat dan konsisten.

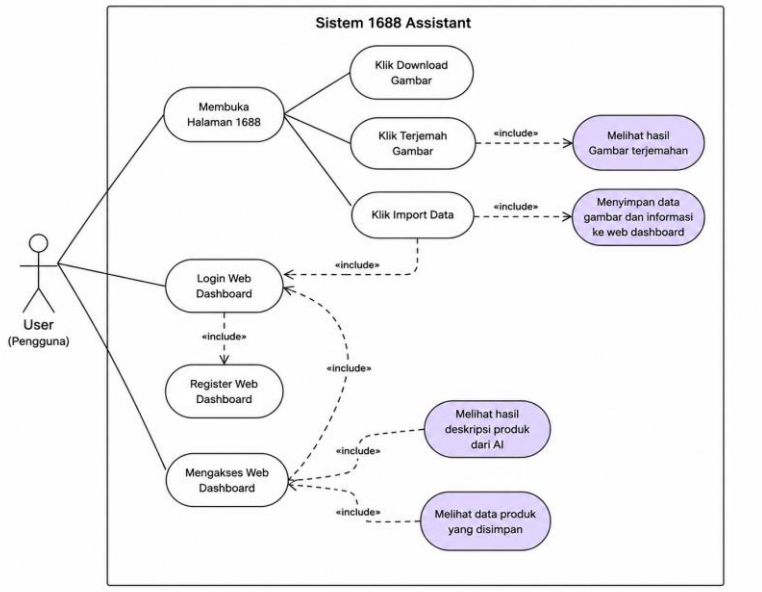
Dengan arsitektur tersebut, proses pengambilan data produk, pengolahan gambar, penyimpanan data, serta pengelolaan informasi produk dapat dilakukan secara terintegrasi melalui Chrome Extension, Backend Flask, Database MySQL, dan Web Dashboard. Pendekatan ini memungkinkan proses penyusunan katalog produk menjadi lebih efisien dibandingkan proses manual karena seluruh tahapan pengolahan dilakukan dalam satu sistem yang saling terhubung.

3.5.1 *Use Case Diagram*

Use Case Diagram digunakan untuk menggambarkan hubungan antara aktor dengan fungsi-fungsi yang disediakan oleh sistem. Diagram ini menunjukkan

layanan yang dapat diakses oleh pengguna tanpa menjelaskan proses internal sistem.

Pada sistem yang diusulkan, terdapat satu aktor utama, yaitu Pelaku Usaha, yang menggunakan sistem untuk memperoleh dan mengelola informasi produk dari platform 1688.com melalui Chrome Extension dan Web Dashboard. Melalui Chrome Extension, pelaku usaha dapat melakukan pengambilan informasi produk (*DOM scraping*), menerjemahkan teks pada gambar produk menggunakan OCR dan Google Translate API, melakukan *image text swapping*, mengunduh gambar asli produk, serta menghasilkan nama produk, deskripsi produk, dan kata kunci menggunakan Gemini. Selain itu, melalui Web Dashboard pengguna dapat mengelola dan meninjau data produk yang telah diproses oleh sistem.



Gambar 3.5 *Use Case Diagram*

Dengan adanya *use case* diagram tersebut, dapat diketahui bahwa seluruh fungsi sistem berpusat pada kebutuhan pelaku usaha dalam memperoleh dan mengelola informasi produk secara otomatis dari platform 1688.com.

3.5.2 *Activity Diagram*

Activity Diagram digunakan untuk menggambarkan alur aktivitas yang terjadi pada sistem ketika pengguna menjalankan suatu fitur hingga sistem menghasilkan keluaran. Diagram ini menunjukkan urutan aktivitas, percabangan keputusan (*decision*), serta interaksi antara pengguna, Ekstensi Chrome, Backend Flask, dan Database MySQL dalam menjalankan proses bisnis sistem.

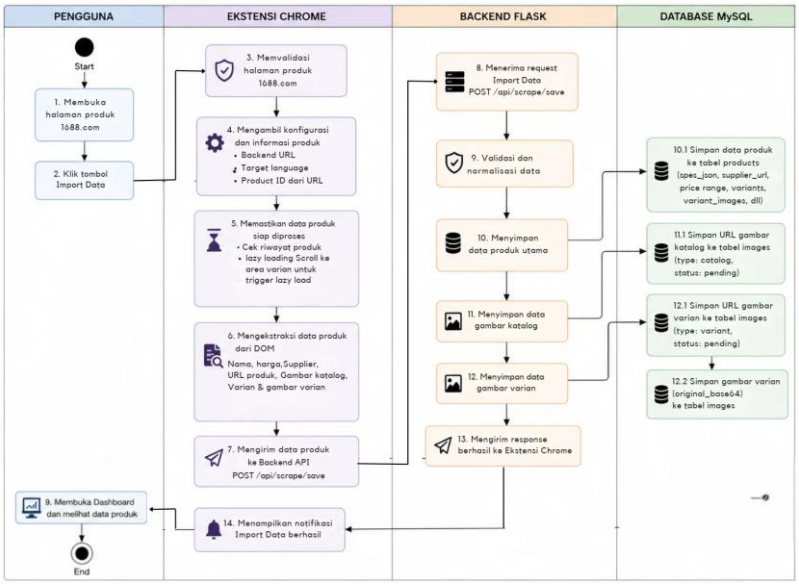
Pada penelitian ini, Activity Diagram disusun berdasarkan fitur-fitur utama yang tersedia pada sistem. Pendekatan ini dipilih agar setiap proses bisnis dapat dijelaskan secara lebih spesifik sesuai dengan implementasi masing-masing fitur. Activity Diagram yang disajikan meliputi proses Import Data Produk, Terjemah Gambar, Download Gambar, Generate Deskripsi Produk, dan Pengaturan Bahasa (Settings).

Pada penelitian ini, Activity Diagram difokuskan pada dua proses utama, yaitu proses pengambilan data produk dari platform 1688.com dan proses OCR serta translasi gambar produk.

a. *Activity Diagram* Fitur Import Data Produk

Activity Diagram ini menggambarkan alur proses penyimpanan data produk dari platform 1688.com ke dalam sistem. Proses diawali ketika pengguna membuka halaman detail produk pada platform 1688.com menggunakan Google Chrome yang telah dipasang ekstensi. Setelah pengguna menekan tombol Import Data, ekstensi melakukan proses DOM Scraping untuk mengambil informasi produk, kemudian mengirimkan data tersebut ke Backend Flask melalui REST API.

Backend menerima dan memvalidasi data yang dikirimkan. Apabila data valid, backend melakukan normalisasi informasi produk, menyimpan data produk ke basis data, serta menyimpan informasi gambar yang berkaitan dengan produk. Setelah seluruh proses penyimpanan selesai, backend mengirimkan respons berhasil kepada ekstensi sehingga pengguna memperoleh notifikasi bahwa data produk berhasil diimpor ke dalam sistem.

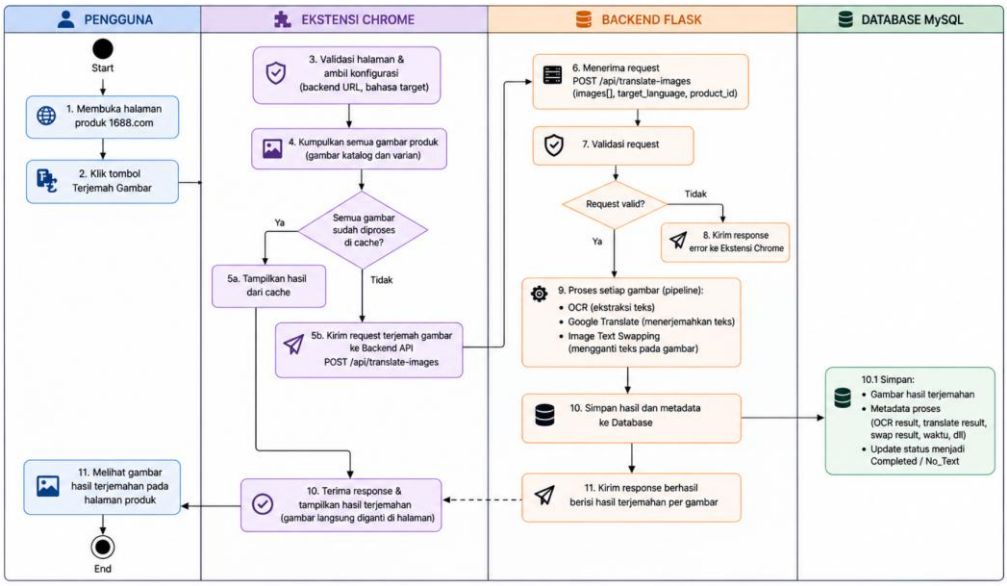


Gambar 3.6 Activity Diagram Fitur Import Data Produk

b. *Activity Diagram* Fitur Terjemah Gambar

Activity Diagram ini menggambarkan proses penerjemahan gambar produk yang mengandung teks berbahasa Mandarin. Proses dimulai ketika pengguna menekan tombol Terjemah Gambar pada ekstensi. Ekstensi kemudian mengirimkan daftar gambar produk ke Backend Flask untuk diproses.

Backend menjalankan tahapan OCR untuk mengekstraksi teks dari gambar, menerjemahkan hasil ekstraksi menggunakan Google Translate API, kemudian melakukan proses *Image Text Swapping* untuk mengganti teks asli pada gambar dengan hasil terjemahan. Setelah proses selesai, gambar hasil terjemahan beserta metadata disimpan ke dalam basis data. Selanjutnya backend mengirimkan respons kepada ekstensi sehingga gambar hasil terjemahan dapat ditampilkan kepada pengguna.

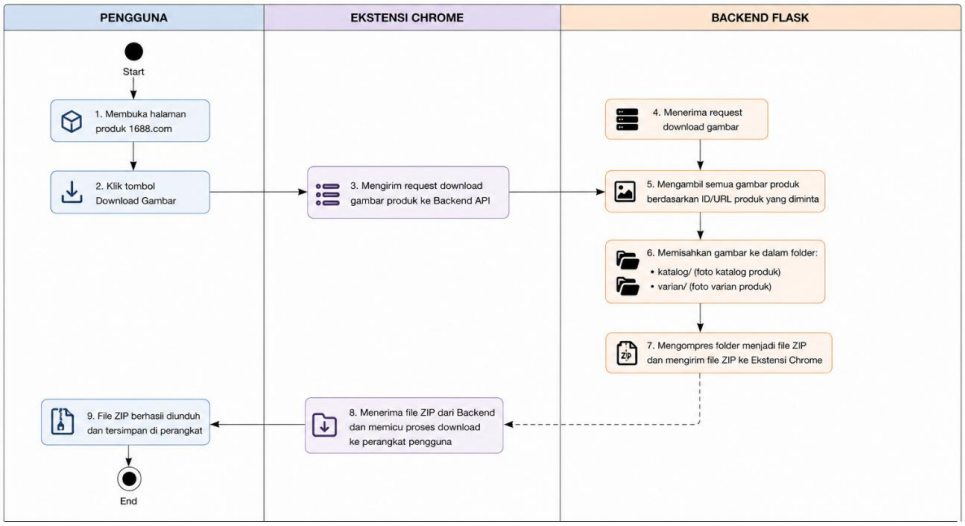


Gambar 3.7 Activity Diagram Fitur Terjemah Gambar

c. *Activity Diagram* Fitur Download Gambar

Activity Diagram ini menggambarkan proses pengunduhan gambar produk. Proses diawali ketika pengguna memilih fitur Download Gambar pada ekstensi. Selanjutnya ekstensi mengirimkan permintaan pengunduhan ke Backend Flask.

Backend mengumpulkan seluruh gambar produk berdasarkan URL yang diterima, memisahkan gambar katalog dan gambar varian, kemudian mengompres seluruh gambar ke dalam berkas ZIP. Setelah proses kompresi selesai, backend mengirimkan berkas ZIP kepada ekstensi sehingga browser dapat memulai proses pengunduhan dan menyimpan file ke perangkat pengguna.

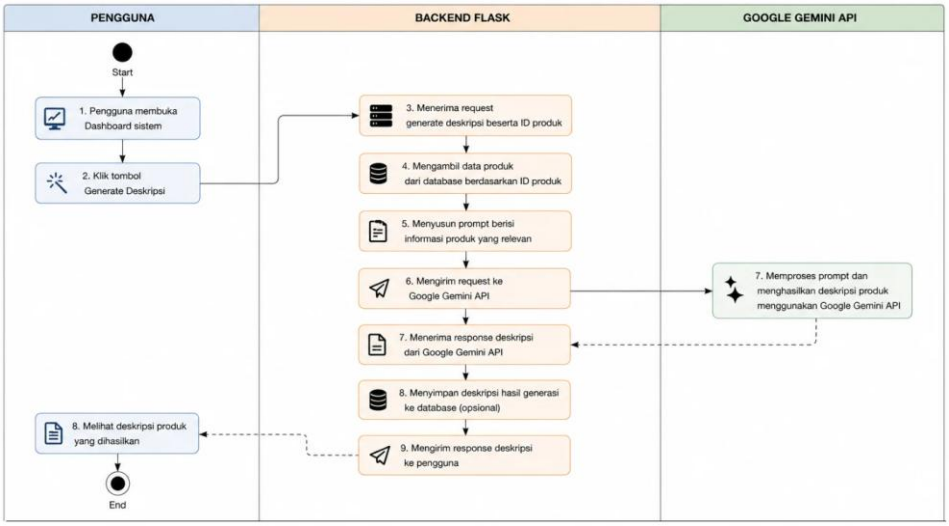


Gambar 3.8 *Activity Diagram* Fitur *Download* Gambar

d. *Activity Diagram* Fitur *Generate* Deskripsi Produk

Activity Diagram ini menggambarkan proses pembuatan deskripsi produk menggunakan layanan Gemini AI. Proses dimulai ketika pengguna memilih salah satu produk pada Web Dashboard dan menekan tombol Generate Deskripsi.

Backend mengambil informasi produk dari basis data kemudian mengirimkan data tersebut ke layanan Gemini AI untuk menghasilkan nama produk, deskripsi, dan kata kunci. Hasil yang diperoleh selanjutnya disimpan kembali ke dalam basis data dan ditampilkan pada Web Dashboard sehingga dapat digunakan sebagai informasi tambahan dalam penyusunan katalog produk.

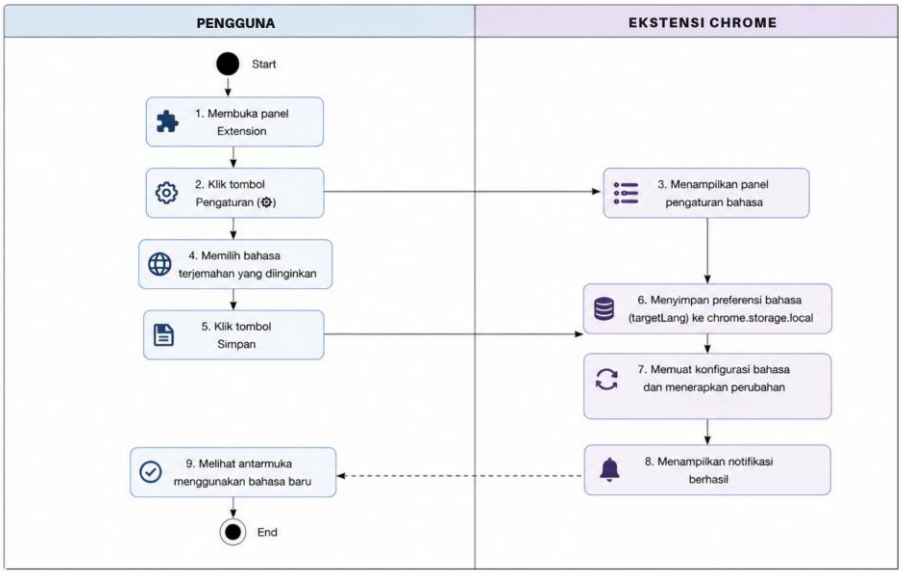


Gambar 3.9 *Activity Diagram* Fitur *Generate* Deskripsi Produk

e. *Activity Diagram* Fitur Pengaturan Bahasa (Settings)

Activity Diagram ini menggambarkan proses pengaturan bahasa tujuan pada ekstensi. Pengguna membuka menu Settings, kemudian memilih bahasa yang diinginkan sebagai bahasa target penerjemahan.

Setelah pengguna menekan tombol Simpan, konfigurasi bahasa disimpan pada penyimpanan lokal (*Chrome Storage*). Selanjutnya antarmuka ekstensi diperbarui secara otomatis sesuai dengan bahasa yang dipilih sehingga seluruh proses penerjemahan berikutnya menggunakan konfigurasi bahasa tersebut.



Gambar 3.10 *Activity Diagram* Fitur Pengaturan Bahasa (*Settings*)

3.5.3 Sequence Diagram

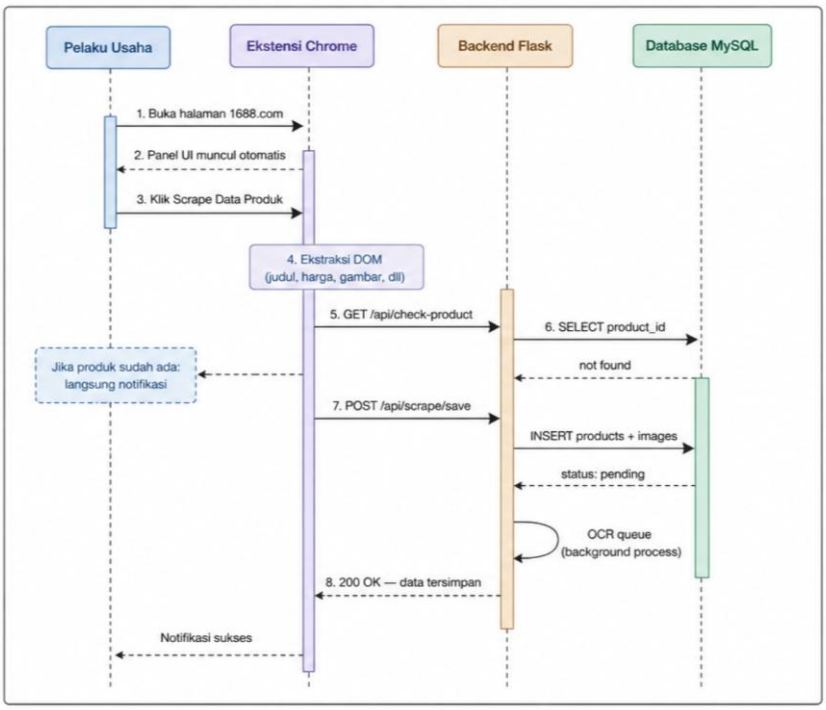
Sequence Diagram digunakan untuk menggambarkan urutan komunikasi antar objek berdasarkan dimensi waktu selama menjalankan suatu proses. Diagram ini memperlihatkan pesan (*message*) yang dikirim antar komponen sistem sehingga hubungan antar komponen dapat dipahami secara lebih jelas.

Pada penelitian ini, *Sequence Diagram* digunakan untuk memodelkan dua proses utama, yaitu komunikasi selama proses pengambilan data produk serta komunikasi selama proses OCR dan translasi gambar.

a. Sequence Diagram Proses Pengambilan Data Produk

Sequence Diagram ini menggambarkan interaksi antara pelaku usaha, ekstensi Google Chrome, backend Flask, dan basis data selama proses scraping berlangsung. Proses dimulai ketika pengguna membuka halaman produk dan menjalankan proses scraping melalui ekstensi. Ekstensi kemudian mengekstraksi

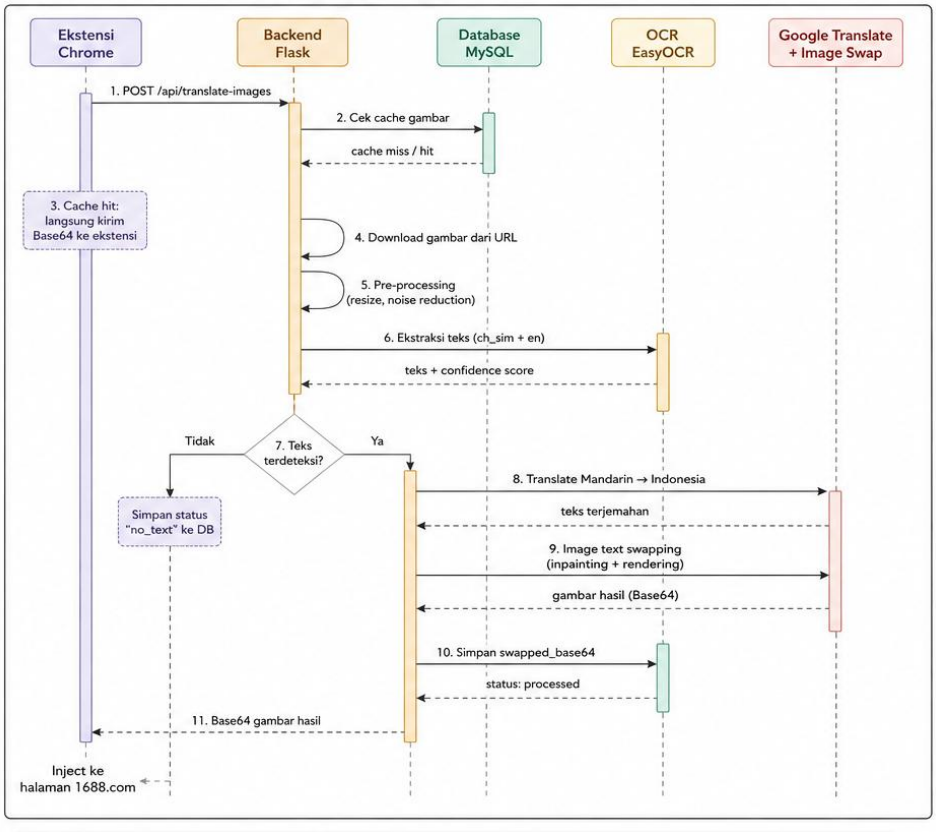
data dari DOM dan mengirimkannya ke backend untuk dilakukan validasi dan penyimpanan data. Setelah proses penyimpanan selesai, backend mengirimkan respons ke ekstensi sehingga pengguna memperoleh notifikasi bahwa proses berhasil dilakukan. Pipeline *OCR* dijalankan sebagai proses latar belakang setelah data berhasil disimpan.



Gambar 3.11 *Sequence Diagram* pengambilan data produk

b. Sequence Diagram Proses OCR dan Translasi

Sequence Diagram ini menggambarkan komunikasi antara backend Flask, basis data, modul OCR, dan layanan translasi selama proses pengolahan gambar berlangsung. Backend terlebih dahulu memeriksa ketersediaan hasil pada cache. Apabila hasil belum tersedia, backend mengunduh gambar, melakukan *pre-processing*, menjalankan OCR menggunakan EasyOCR, menerjemahkan teks hasil ekstraksi, kemudian melakukan proses image text swapping. Hasil akhir berupa gambar yang telah diterjemahkan disimpan ke dalam basis data dan dikirimkan kembali ke ekstensi untuk ditampilkan pada halaman produk.



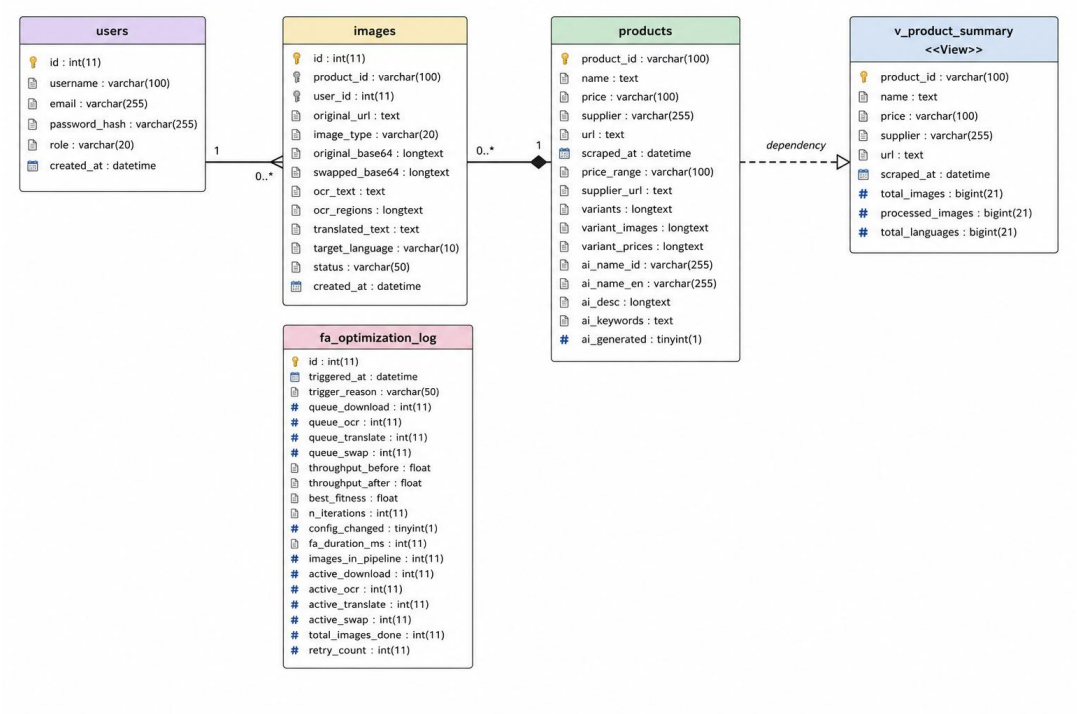
Gambar 3.12 *Sequence Diagram OCR* dan Translasi

3.5.4 Perancangan Basis Data

Perancangan basis data dilakukan untuk mendukung proses penyimpanan, pengelolaan, dan pengambilan data yang dihasilkan selama sistem berjalan. Basis data pada penelitian ini menggunakan MySQL sebagai *Database Management System* (DBMS) karena mampu mengelola data relasional secara efisien, mendukung integrasi dengan framework Flask, serta memiliki performa yang baik dalam menangani proses penyimpanan dan pengambilan data.

Basis data dirancang untuk menyimpan informasi produk hasil pengambilan data dari platform 1688.com, hasil pengolahan gambar menggunakan *Optical Character Recognition* (OCR), hasil penerjemahan menggunakan Google Translate API, hasil *Image Text Swapping*, informasi produk yang dihasilkan melalui Gemini AI, data pengguna, serta log proses optimasi Firefly Algorithm. Struktur basis data disusun agar setiap informasi saling terhubung sehingga memudahkan proses pengelolaan data, menjaga konsistensi informasi, serta meminimalkan redundansi data.

Perancangan struktur basis data direpresentasikan menggunakan Class Diagram yang ditunjukkan pada Gambar 3.13.



Gambar 3.13 Class Diagram Database katalog_1688

Berdasarkan Class Diagram yang telah dirancang, basis data terdiri atas lima kelas utama, yaitu users, products, images, v_product_summary, dan fa_optimization_log. Kelas users digunakan untuk menyimpan informasi pengguna yang memiliki hak akses terhadap sistem. Kelas products berfungsi sebagai penyimpanan utama data produk hasil pengambilan data dari platform 1688.com, meliputi nama produk, harga, pemasok, variasi produk, serta informasi produk yang dihasilkan melalui Gemini AI, seperti nama produk, deskripsi, dan kata kunci.

Kelas images digunakan untuk menyimpan informasi gambar produk, meliputi URL gambar asli, hasil OCR, hasil penerjemahan, hasil *Image Text Swapping*, serta status proses pengolahan gambar. Setiap produk dapat memiliki lebih dari satu gambar sehingga hubungan antara kelas products dan images bersifat one-to-many. Selain itu, setiap data gambar dikaitkan dengan pengguna melalui atribut user_id, sehingga satu pengguna dapat memiliki banyak data gambar hasil pengolahan.

Sistem juga memanfaatkan v_product_summary sebagai *database view* yang digunakan untuk menyajikan ringkasan informasi produk, seperti jumlah gambar, jumlah gambar yang telah diproses, serta jumlah bahasa hasil penerjemahan. Informasi tersebut dimanfaatkan untuk mendukung penyajian data pada Web Dashboard tanpa perlu melakukan proses agregasi secara berulang.

Selain itu, kelas fa_optimization_log digunakan untuk menyimpan riwayat proses optimasi yang dilakukan oleh Firefly Algorithm, meliputi penyebab optimasi dijalankan, konfigurasi pipeline sebelum dan sesudah optimasi, nilai *fitness*, serta informasi performa proses. Data tersebut digunakan sebagai dokumentasi sekaligus bahan evaluasi terhadap kinerja optimasi pipeline yang diterapkan pada sistem.

Perancangan hubungan antar kelas dilakukan untuk menjaga integritas data, mengurangi redundansi penyimpanan, serta mendukung proses pengolahan, penyajian informasi produk, dan evaluasi kinerja sistem secara efisien.

3.5.5 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk merancang media interaksi antara pelaku usaha dengan sistem sehingga seluruh fungsi dapat digunakan secara mudah, efisien, dan intuitif. Antarmuka sistem dirancang dengan mempertimbangkan alur kerja pengadaan produk dari platform 1688.com, sehingga pengguna dapat melakukan proses DOM scraping, translasi gambar, pengelolaan data produk, serta pemanfaatan fitur berbasis Artificial Intelligence melalui tampilan yang sederhana dan mudah dipahami.

Pada penelitian ini, antarmuka sistem terdiri atas dua komponen utama, yaitu ekstensi Google Chrome dan Web Dashboard. Ekstensi Google Chrome berfungsi sebagai media untuk melakukan proses DOM scraping, menerjemahkan teks pada gambar produk melalui proses OCR, Google Translate API, dan image text swapping, serta mengirimkan data hasil pengolahan ke backend. Sementara itu, Web Dashboard berfungsi sebagai media untuk mengelola data produk hasil scraping, menampilkan informasi produk yang telah diproses, serta menyediakan fasilitas pencarian dan pengelolaan data produk.

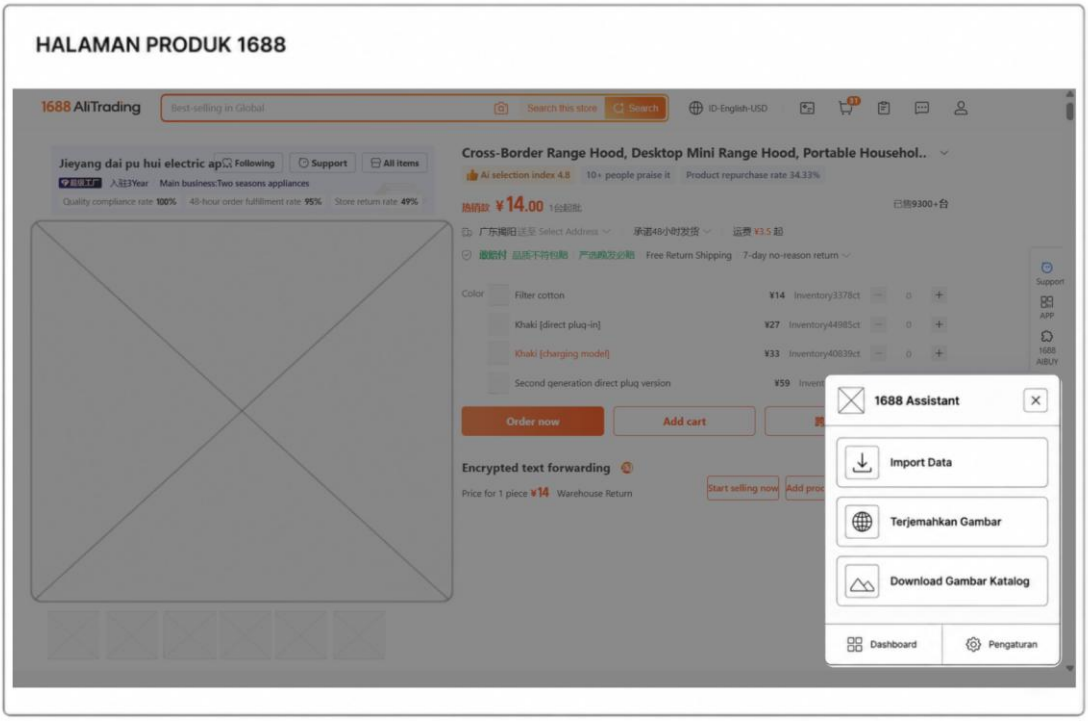
3.5.5.1 Perancangan Antarmuka Ekstensi Google Chrome

Perancangan antarmuka ekstensi Google Chrome bertujuan untuk menyediakan media interaksi bagi pengguna dalam melakukan proses pengambilan dan pengolahan data produk secara langsung pada halaman produk 1688.com. Antarmuka ekstensi dirancang agar muncul setelah pengguna mengaktifkan ekstensi pada halaman produk, sehingga seluruh fungsi utama dapat diakses tanpa harus berpindah ke aplikasi lain.

Antarmuka ekstensi menyediakan tiga fitur utama, yaitu Import Data untuk mengambil informasi produk menggunakan teknik DOM scraping, Terjemahkan Gambar untuk menerjemahkan teks berbahasa Mandarin pada gambar melalui proses OCR, Google Translate API, dan image text swapping, serta Download Gambar Katalog untuk mengunduh seluruh gambar produk yang tersedia pada

halaman. Selain itu, ekstensi juga menyediakan akses menuju Dashboard dan Pengaturan sebagai navigasi tambahan bagi pengguna.

Rancangan antarmuka pada Gambar 3.14 memperlihatkan posisi ekstensi yang ditampilkan di sisi kanan halaman produk 1688.com. Penempatan tersebut memungkinkan pengguna menjalankan seluruh proses pengolahan data secara langsung tanpa mengganggu aktivitas penelusuran informasi produk.



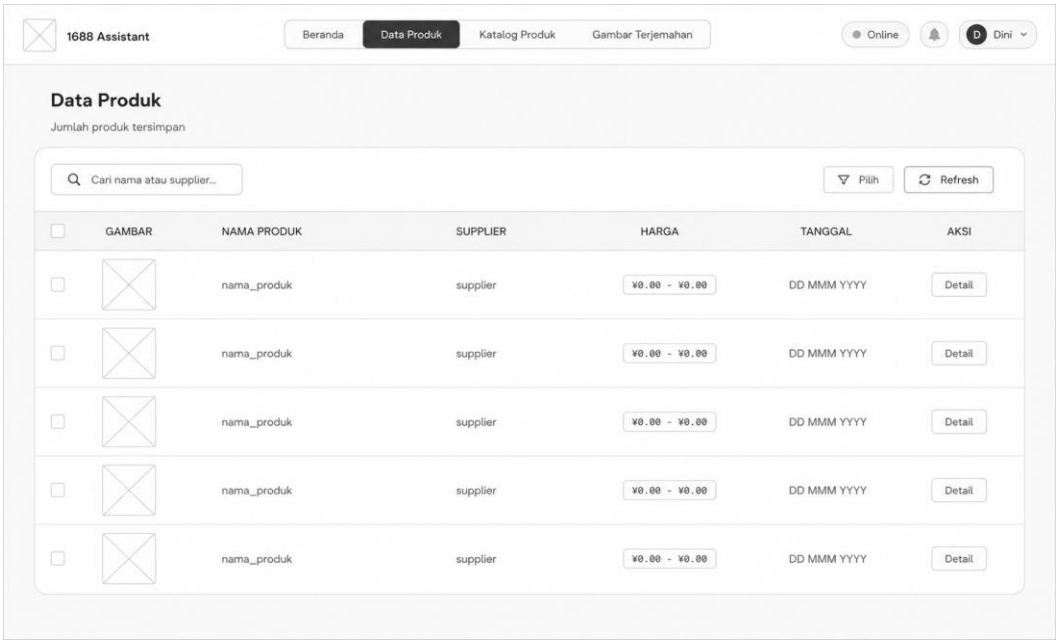
Gambar 3.14 Perancangan Antarmuka Ekstensi Google Chrome

3.5.5.2 Perancangan Antarmuka Dashboard

Web Dashboard dirancang sebagai pusat pengelolaan seluruh data hasil *DOM scraping* yang dikirimkan dari ekstensi Google Chrome. Melalui dashboard, pengguna dapat melihat daftar produk yang telah diproses, melakukan pencarian produk berdasarkan nama atau kata kunci tertentu, serta memantau status pemrosesan setiap produk.

Setiap data produk ditampilkan dalam bentuk tabel yang memuat gambar utama, nama produk, status pemrosesan, tanggal pengolahan, dan aksi untuk melihat informasi produk secara lebih lengkap. Selain itu, dashboard juga menampilkan ringkasan jumlah produk yang telah diproses sehingga pengguna dapat memantau hasil pengolahan secara keseluruhan.

Perancangan antarmuka dashboard dilakukan dengan mengutamakan kemudahan navigasi dan penyajian informasi secara terstruktur sehingga pengguna dapat mengakses hasil pengolahan produk dengan lebih efisien.



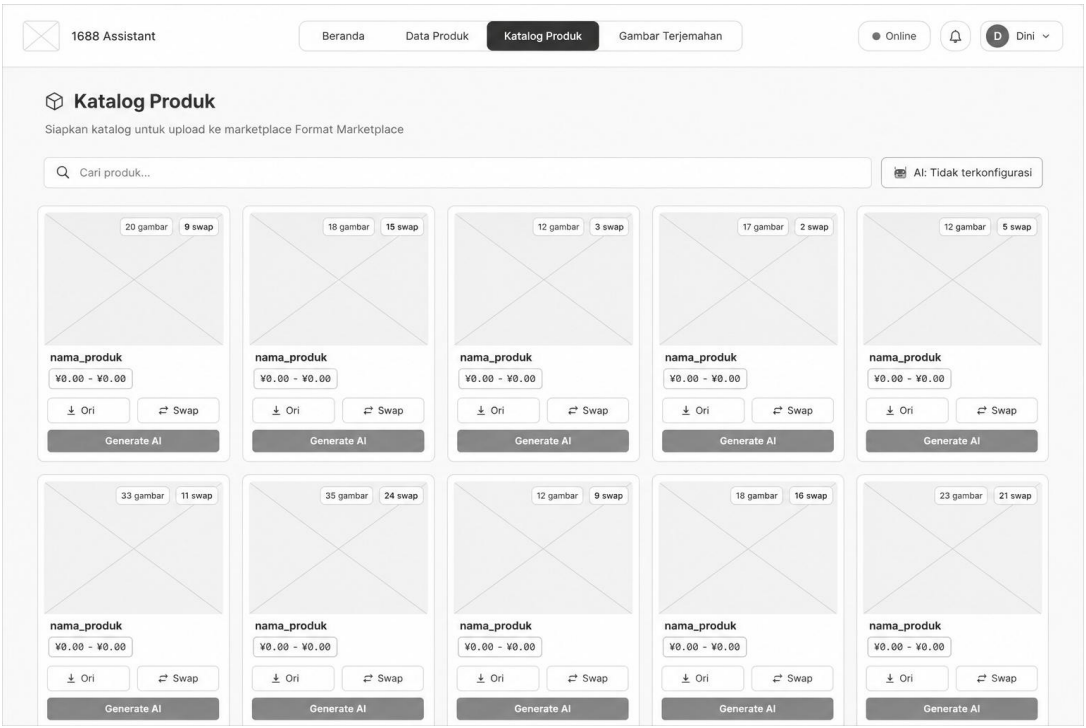
Gambar 3.15 Perancangan Antarmuka Dashboard

3.5.5.3 Perancangan Antarmuka Halaman Detail Produk

Halaman Detail Produk dirancang untuk menampilkan informasi hasil pengolahan dari satu produk yang dipilih pada menu Data Produk. Halaman ini menyediakan informasi produk yang telah dihasilkan oleh sistem, meliputi nama produk, deskripsi produk, dan kata kunci hasil pemrosesan menggunakan Gemini.

Selain informasi tekstual, halaman ini juga menampilkan galeri gambar yang berisi gambar asli (*original image*) dan gambar hasil *image text swapping* sehingga pengguna dapat membandingkan hasil penerjemahan secara langsung. Halaman ini juga menyajikan status setiap tahapan pengolahan, mulai dari proses *scraping*, OCR, penerjemahan, hingga *image text swapping*, sebagai informasi bahwa seluruh proses telah berhasil diselesaikan.

Perancangan halaman ini bertujuan agar seluruh hasil pengolahan produk dapat diakses dalam satu halaman sehingga memudahkan pengguna melakukan verifikasi dan penyusunan katalog produk multibahasa.



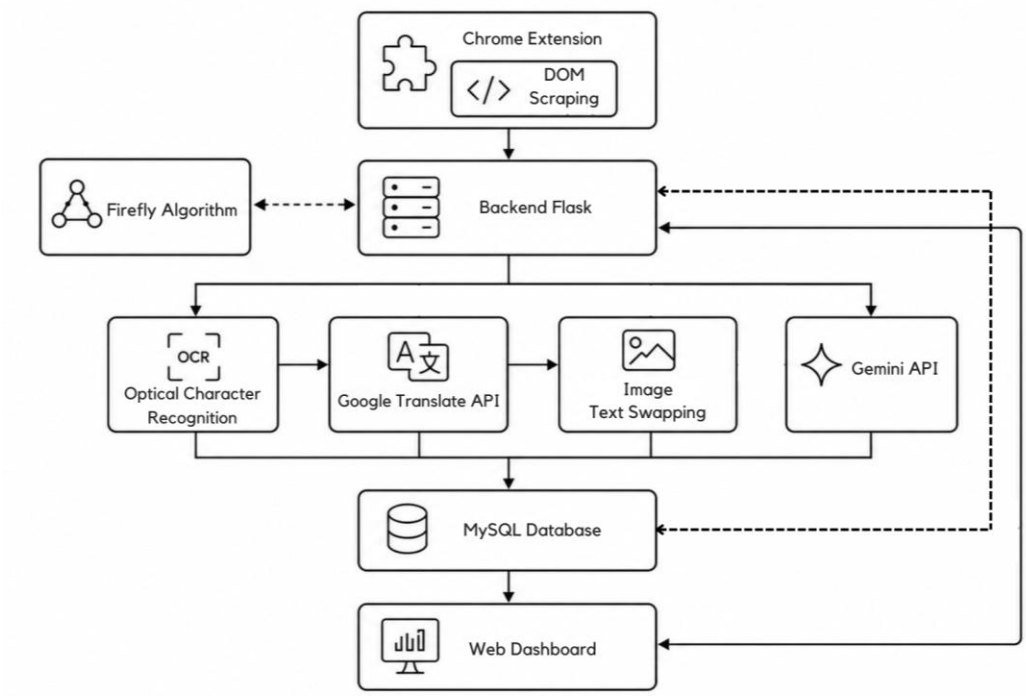
Gambar 3.16 Perancangan Antarmuka Halaman Detail Produk

3.6 Perancangan Modul Sistem

Perancangan modul sistem dilakukan untuk membagi sistem ke dalam beberapa komponen berdasarkan fungsi dan tanggung jawab masing-masing. Pendekatan modular diterapkan agar setiap komponen dapat dikembangkan, dipelihara, dan diuji secara independen tanpa memengaruhi modul lainnya. Selain itu, pembagian modul mempermudah proses integrasi antar komponen melalui mekanisme komunikasi yang terstruktur sehingga aliran data antar modul menjadi lebih jelas.

Pada sistem yang diusulkan, modul utama terdiri atas Chrome Extension, Backend Flask, Modul OCR, Modul Terjemahan, Modul Image Text Swapping, Gemini API, MySQL Database, Web Dashboard, dan Firefly Algorithm. Masing-masing modul memiliki fungsi yang saling melengkapi dalam membentuk alur pemrosesan sistem, mulai dari proses DOM scraping, ekstraksi teks menggunakan OCR, penerjemahan teks, penggantian teks pada gambar (*image text swapping*), optimasi penjadwalan proses menggunakan Firefly Algorithm, hingga penyimpanan dan penyajian hasil melalui Web Dashboard.

Hubungan antar modul sistem ditunjukkan pada Gambar 3.17.



Gambar 3.17 Hubungan Antar Modul Sistem

Berdasarkan Gambar 3.17, setiap modul pada sistem memiliki hubungan yang saling terintegrasi dalam mendukung proses pengolahan data. Chrome Extension berkomunikasi dengan Backend Flask untuk mengirimkan data hasil *DOM scraping* sekaligus menerima respons dari backend. Backend Flask berperan sebagai pusat koordinasi yang menghubungkan seluruh modul sehingga pertukaran data antar komponen dapat berlangsung secara terstruktur.

Dalam menjalankan proses pengolahan gambar, Backend Flask terhubung dengan Modul Optical Character Recognition (OCR), Google Translate API, dan Modul Image Text Swapping. Ketiga modul tersebut saling berhubungan membentuk rangkaian pengolahan gambar, di mana hasil dari suatu modul menjadi masukan bagi modul berikutnya. Selain itu, Backend Flask juga terhubung dengan Gemini API untuk menyediakan layanan pembuatan ringkasan informasi produk.

Hubungan antara Firefly Algorithm dan Backend Flask bersifat dua arah. Firefly Algorithm memberikan hasil optimasi parameter penjadwalan proses kepada Backend Flask, sedangkan Backend Flask menyediakan informasi yang diperlukan sebagai dasar evaluasi proses optimasi. Dengan demikian, keputusan penjadwalan yang dihasilkan dapat diterapkan pada proses pengolahan data di dalam sistem.

Seluruh hasil pemrosesan dari berbagai modul disimpan pada MySQL Database sebagai media penyimpanan terpusat. Selanjutnya, Web Dashboard

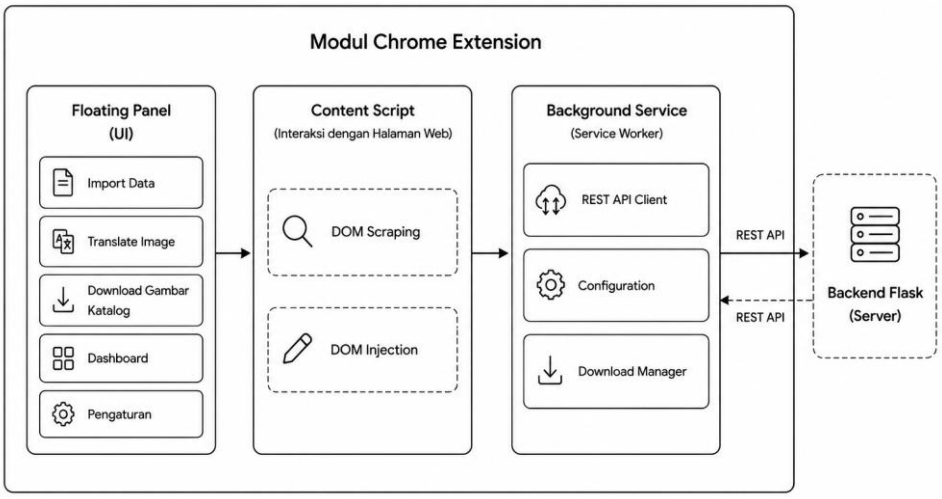
terhubung dengan MySQL Database untuk menampilkan data kepada pengguna. Selain itu, Web Dashboard juga berkomunikasi langsung dengan Backend Flask ketika pengguna menjalankan suatu fungsi yang memerlukan pemrosesan sistem, sehingga Backend Flask dapat mengakses maupun memperbarui data pada database sesuai kebutuhan.

3.6.1 Modul Chrome Extension

Modul Chrome Extension merupakan komponen *client-side* yang dirancang sebagai media interaksi antara pengguna dengan sistem pada platform 1688.com. Modul ini memungkinkan pengguna mengakses layanan sistem secara langsung melalui halaman produk tanpa perlu berpindah ke aplikasi lain. Selain berfungsi sebagai antarmuka pengguna, modul ini juga dirancang untuk menghubungkan halaman web dengan Backend Flask sehingga proses pengiriman data dan penerimaan hasil pemrosesan dapat dilakukan secara terintegrasi.

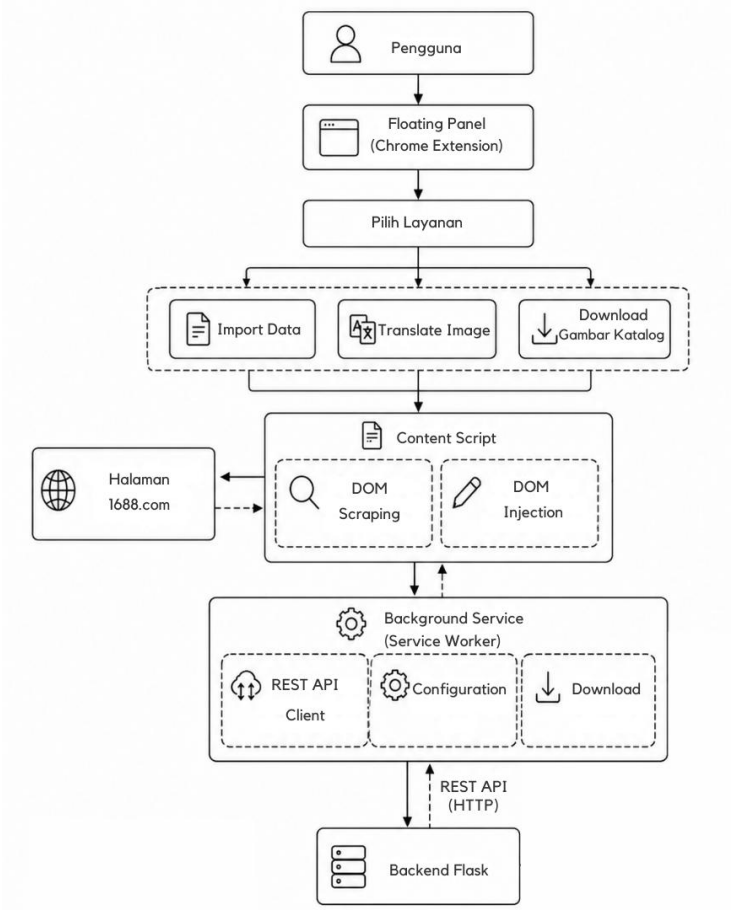
Secara umum, Modul Chrome Extension terdiri atas tiga komponen utama, yaitu Floating Panel, Content Script, dan Background Service. Floating Panel berfungsi sebagai antarmuka pengguna yang menyediakan akses terhadap layanan sistem. Content Script bertugas berinteraksi secara langsung dengan struktur halaman web (*Document Object Model* atau DOM), sedangkan Background Service bertugas mengelola komunikasi antara Chrome Extension dengan Backend Flask melalui REST API. Pembagian komponen tersebut dirancang agar setiap komponen memiliki tanggung jawab yang jelas sehingga proses interaksi pengguna, komunikasi sistem, dan pengelolaan data dapat dilakukan secara terpisah namun tetap saling terintegrasi.

Pada perancangannya, Content Script memanfaatkan mekanisme DOM Scraping dan DOM Injection sebagai bagian dari proses interaksi dengan halaman web. DOM Scraping digunakan untuk memperoleh informasi produk dengan mengakses elemen-elemen HTML pada halaman 1688.com, sedangkan DOM Injection digunakan untuk menampilkan kembali hasil pemrosesan yang diterima dari Backend Flask ke dalam halaman web. Dengan mekanisme tersebut, proses pengambilan data maupun penyajian hasil dapat dilakukan secara langsung pada halaman produk tanpa mengubah struktur utama halaman.



Gambar 3.18 Perancangan Modul Chrome Extension

Gambar 3.18 menunjukkan perancangan Modul Chrome Extension yang terdiri atas tiga komponen utama, yaitu Floating Panel, Content Script, dan Background Service. Floating Panel menyediakan antarmuka bagi pengguna untuk mengakses layanan sistem, seperti Import Data, Translate Image, Download Gambar Katalog, Dashboard, dan Pengaturan. Permintaan dari pengguna kemudian diteruskan ke Content Script untuk berinteraksi dengan halaman web melalui mekanisme DOM Scraping dan DOM Injection. Selanjutnya, Background Service berperan sebagai penghubung antara Chrome Extension dan Backend Flask melalui komunikasi REST API, sehingga proses pertukaran data antara *client-side* dan *server-side* dapat dilakukan secara terintegrasi.



Gambar 3.19 Alur Kerja Modul Chrome Extension

Gambar 3.19 menunjukkan alur kerja Modul Chrome Extension. Proses dimulai ketika pengguna memilih salah satu layanan melalui Floating Panel. Permintaan tersebut diteruskan ke Content Script, yang kemudian berinteraksi dengan halaman web menggunakan mekanisme DOM Scraping untuk memperoleh data yang dibutuhkan. Data hasil scraping selanjutnya dikirimkan ke Background Service, kemudian diteruskan ke Backend Flask melalui REST API untuk diproses sesuai dengan layanan yang dipilih pengguna. Setelah proses pada Backend Flask selesai, hasil pemrosesan dikirim kembali ke Background Service dan diteruskan ke Content Script. Selanjutnya, DOM Injection digunakan untuk memperbarui elemen pada halaman 1688.com sehingga hasil pemrosesan dapat ditampilkan secara langsung kepada pengguna.

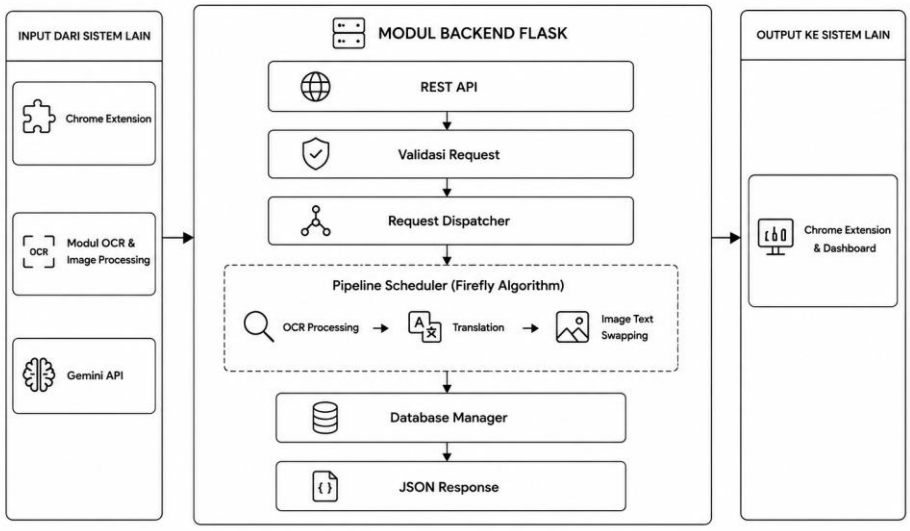
3.6.2 Modul Backend Flask

Modul Backend Flask berfungsi sebagai layanan backend yang menerima permintaan dari Chrome Extension maupun Web Dashboard, kemudian mengkoordinasikan seluruh proses pengolahan data pada sistem. Backend menyediakan layanan REST API sebagai media komunikasi antar komponen

sistem, melakukan validasi data yang diterima, serta mengelola proses penyimpanan dan pengambilan data pada basis data.

Selain itu, Backend Flask bertugas mengatur alur pemrosesan menggunakan Pipeline Scheduler yang dioptimalkan dengan Firefly Algorithm, kemudian mengkoordinasikan proses OCR, Translation, dan Image Text Swapping sesuai dengan layanan yang dipilih pengguna. Backend juga berkomunikasi dengan Gemini API untuk menghasilkan nama, deskripsi, dan kata kunci produk sebelum mengirimkan hasil pemrosesan kembali kepada Chrome Extension maupun Web Dashboard melalui REST API.

Alur kerja implementasi Modul Backend Flask ditunjukkan pada Gambar 3.20, sedangkan spesifikasi modul disajikan pada Tabel 3.6.



Gambar 3.20 Alur Implementasi Modul Chrome Extension

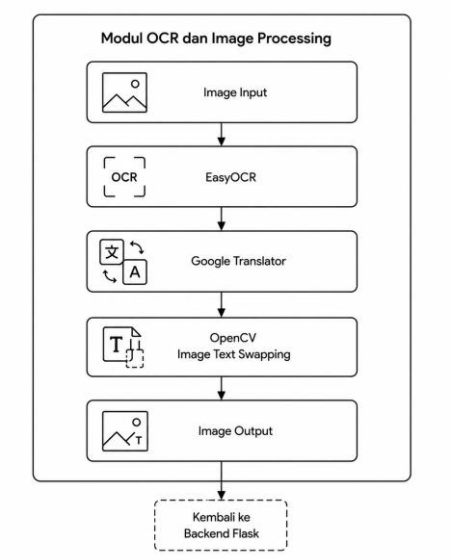
Gambar 3.20 menunjukkan proses dimulai ketika backend menerima permintaan melalui REST API dari Chrome Extension atau Web Dashboard. Selanjutnya backend melakukan validasi request untuk memastikan data yang diterima sesuai dengan kebutuhan sistem. Request kemudian diteruskan ke *Requestdispatcher* untuk menentukan proses yang akan dijalankan berdasarkan endpoint yang diakses. Selanjutnya Pipeline Scheduler yang dioptimalkan menggunakan Firefly Algorithm mengatur urutan pemrosesan OCR, Translation, dan Image Text Swapping. Setelah seluruh proses selesai, data disimpan atau diambil melalui Database Manager, kemudian backend mengembalikan hasil pemrosesan dalam bentuk JSON Response kepada Chrome Extension maupun Web Dashboard melalui REST API.

Tabel 3.6 Spesifikasi Modul Backend Flask

Komponen	Deskripsi
Tujuan	Mengoordinasikan proses pengolahan data, mengelola komunikasi antar modul, serta menyediakan layanan REST API bagi Chrome Extension dan Web Dashboard.
Input	Data hasil DOM scraping dari Chrome Extension dan permintaan dari Web Dashboard.
Proses	REST API → Validasi Request → Request Dispatcher → Pipeline Scheduler (Firefly Algorithm) → OCR → Translation → Image Text Swapping → Database → JSON Response.
Output	Data produk yang telah diproses, hasil OCR, hasil translasi gambar, deskripsi produk, serta respons REST API kepada Chrome Extension dan Web Dashboard.

3.6.3 Modul OCR dan Image Processing

Modul OCR dan Image Processing berfungsi untuk melakukan ekstraksi teks dari gambar produk, menerjemahkan hasil ekstraksi ke bahasa tujuan, serta menghasilkan gambar baru melalui proses Image Text Swapping. Pada penelitian ini, proses ekstraksi teks dirancang menggunakan EasyOCR karena mendukung pengenalan teks multibahasa, termasuk karakter Mandarin yang umum digunakan pada gambar produk platform 1688. Selanjutnya, teks hasil ekstraksi diterjemahkan ke bahasa tujuan menggunakan modul Google Translator sebelum diterapkan kembali ke gambar menggunakan proses Image Text Swapping berbasis OpenCV. Hasil pemrosesan kemudian dikembalikan kepada Backend Flask untuk diteruskan kepada Chrome Extension maupun Web Dashboard.



Gambar 3.21 Perancangan Modul OCR dan Image Processing

Gambar 3.21 menunjukkan alur kerja Modul OCR dan Image Processing. Proses dimulai dengan menerima gambar produk sebagai masukan. Selanjutnya, EasyOCR melakukan ekstraksi teks dari gambar, kemudian hasil ekstraksi diterjemahkan menggunakan Google Translator ke bahasa yang dipilih pengguna. Teks hasil translasi selanjutnya digunakan pada proses Image Text Swapping berbasis OpenCV untuk menghasilkan gambar baru dengan teks yang telah diterjemahkan. Hasil akhir modul berupa gambar produk yang telah diperbarui dan dikembalikan kepada Backend Flask untuk diproses lebih lanjut.

Tabel 3.7 Spesifikasi Modul OCR dan Image Processing

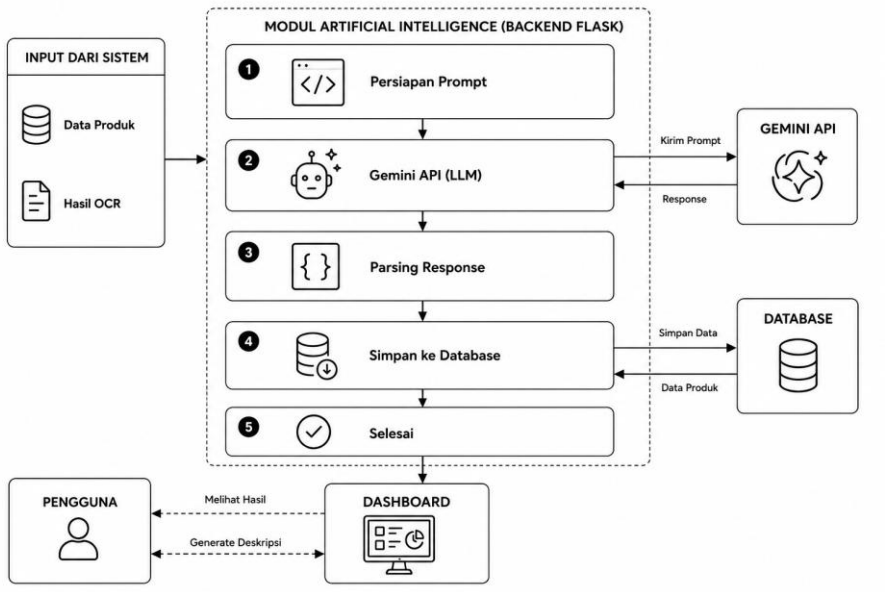
Komponen	Deskripsi
Tujuan	Mengekstraksi teks dari gambar, menerjemahkan teks ke bahasa tujuan, serta menghasilkan gambar hasil *Image Text Swapping*.
Input	Gambar produk yang diterima dari Backend Flask.
Proses	EasyOCR → Google Translator → OpenCV Image Text Swapping.
Output	Gambar hasil *Image Text Swapping*, teks hasil OCR, dan teks hasil translasi yang dikembalikan ke Backend Flask.

3.6.4 Modul Artificial Intelligence

Modul Artificial Intelligence digunakan untuk menghasilkan informasi tambahan berupa nama produk, deskripsi produk, dan kata kunci secara otomatis berdasarkan informasi produk hasil DOM scraping dan teks hasil OCR yang telah diterjemahkan. Modul ini memanfaatkan Gemini API sebagai *Large Language Model* (LLM) untuk menghasilkan konten yang lebih informatif dan sesuai dengan konteks produk.

Proses dimulai dengan penyusunan *prompt* berdasarkan data produk dan hasil OCR yang telah diterjemahkan. Selanjutnya, *prompt* dikirimkan ke Gemini API untuk menghasilkan nama produk, deskripsi produk, dan kata kunci. Respons yang diterima kemudian diproses oleh Backend Flask, disimpan ke dalam basis data, dan ditampilkan pada Web Dashboard sebagai informasi tambahan yang dapat membantu pengguna memahami karakteristik produk serta mendukung proses pencarian dan pengelompokan produk.

Alur kerja Modul Artificial Intelligence ditunjukkan pada Gambar 3.22, sedangkan spesifikasi modul disajikan pada Tabel 3.8.



Gambar 3.22 Alur Implementasi Artificial Intelligence

Gambar 3.22 menunjukkan alur implementasi Modul Artificial Intelligence. Proses dimulai ketika Backend Flask menerima informasi produk hasil DOM scraping dan teks hasil OCR yang telah diterjemahkan sebagai masukan untuk penyusunan *prompt*. Selanjutnya, *prompt* dikirimkan ke Gemini API untuk menghasilkan nama produk, deskripsi produk, dan kata kunci. Respons yang diterima kemudian diproses oleh Backend Flask sebelum disimpan ke dalam basis

data. Hasil akhir selanjutnya ditampilkan pada Web Dashboard sehingga dapat digunakan sebagai informasi tambahan dalam penyusunan katalog produk.

Tabel 3.8 Spesifikasi Modul Artificial Intelligence

Komponen	Deskripsi
Tujuan	Menghasilkan nama produk, deskripsi produk, dan kata kunci secara otomatis menggunakan Gemini API.
Input	Informasi produk hasil DOM scraping dan teks hasil OCR yang telah diterjemahkan.
Proses	Penyusunan *Prompt* → Pengiriman Permintaan ke Gemini API → Parsing Respons → Penyimpanan Hasil ke Basis Data.
Output	Nama produk, deskripsi produk, dan kata kunci yang dihasilkan oleh Gemini API serta disimpan ke dalam basis data.

3.6.5 Modul *Firefly Algorithm*

Modul *Firefly Algorithm* merupakan modul optimasi yang dirancang untuk menentukan konfigurasi *Pipeline Scheduler* yang sesuai dengan kondisi pemrosesan gambar. Pada penelitian ini, *Firefly Algorithm* digunakan untuk mengoptimalkan jumlah *Download Worker*, *OCR Worker*, *Translate Worker*, dan *Image Text Swapping Worker* sehingga setiap tahapan *pipeline* dapat berjalan secara efisien sesuai dengan beban kerja yang sedang diproses.

Modul *Firefly Algorithm* menerima masukan berupa informasi performa *pipeline* yang diperoleh dari *Performance Monitor*. Informasi tersebut terdiri atas panjang antrean (*queue*), jumlah *active task*, nilai *throughput*, dan jumlah *retry* pada setiap tahapan *pipeline*. Data tersebut digunakan sebagai dasar evaluasi untuk menentukan apakah konfigurasi *Pipeline Scheduler* masih sesuai atau perlu dilakukan proses optimasi.

Proses optimasi dirancang untuk dijalankan ketika *Performance Monitor* mendeteksi adanya penurunan performa pada *pipeline*. Kondisi tersebut meliputi terjadinya *bottleneck* akibat panjang antrean yang melebihi ambang batas, penurunan nilai *throughput*, peningkatan jumlah *retry*, atau tingginya penggunaan memori sistem. Dengan mekanisme tersebut, proses optimasi hanya dilakukan ketika diperlukan sehingga penggunaan sumber daya sistem tetap efisien.

Pada penelitian ini, setiap kandidat solusi (*firefly*) merepresentasikan satu konfigurasi *Pipeline Scheduler* yang terdiri atas kombinasi jumlah *worker* pada setiap tahapan *pipeline*. Representasi kandidat solusi dinyatakan sebagai berikut.

$$X = [D, O, T, S] \qquad (3.1)$$

dengan:

D = jumlah *Download Worker*
O= jumlah *OCR Worker*
T = jumlah *Translate Worker*
S = jumlah *Image Text Swapping Worker*

Setiap kandidat solusi kemudian dievaluasi menggunakan fungsi *fitness* berdasarkan kondisi aktual *pipeline* yang diperoleh dari *Performance Monitor*. Nilai *fitness* digunakan untuk membandingkan kualitas setiap konfigurasi *Pipeline Scheduler*. Firefly dengan nilai *fitness* yang lebih tinggi dianggap memiliki solusi yang lebih baik sehingga akan menarik *firefly* lain untuk bergerak menuju konfigurasi tersebut hingga diperoleh konfigurasi *Pipeline Scheduler* yang optimal.

3.6.5.1 Fungsi Fitness *Firefly Algorithm*

Fungsi *fitness* digunakan untuk mengevaluasi kualitas setiap kandidat konfigurasi *Pipeline Scheduler* yang dihasilkan oleh *Firefly Algorithm*. Tujuan evaluasi ini adalah menentukan konfigurasi *worker* yang paling sesuai dengan kondisi pemrosesan gambar sehingga proses *pipeline* dapat berjalan secara efisien.

Pada penelitian ini, fungsi *fitness* dirancang berdasarkan empat komponen evaluasi, yaitu Throughput, Parameter Match, Resource Efficiency, dan Retry Rate. Keempat komponen tersebut digunakan untuk menilai kemampuan konfigurasi *worker* dalam menjaga keseimbangan antara kecepatan pemrosesan, kesesuaian kapasitas *worker*, efisiensi penggunaan sumber daya, serta tingkat keberhasilan proses. Indikator penilaian setiap komponen ditunjukkan pada Tabel 3.9.

Tabel 3.9 Indikator Penilaian Komponen Fungsi Fitness

Komponen	Indikator Penilaian
Throughput	Jumlah *task* yang berhasil diproses per satuan waktu.
Parameter Match	Kesesuaian jumlah *worker* terhadap panjang *queue* dan jumlah *active task*.

Komponen	Indikator Penilaian
Resource Efficiency	Tingkat penggunaan *worker* dan memori sistem selama proses berlangsung.
Retry Rate	Jumlah *task* yang mengalami *retry* atau diproses ulang.

Persamaan fungsi *fitness* yang digunakan pada penelitian ini ditunjukkan pada Persamaan (3.1).

$$Fitness = w_1 F_{throughput} + w_2 F_{match} + w_3 F_{resource} + w_4 F_{retry} \quad (3.2)$$

dengan ketentuan

$$w_1 + w_2 + w_3 + w_4 = 1 \quad (3.3)$$

Keterangan:

$F_{throughput}$ = Nilai evaluasi *throughput*.

F_{match} = Nilai kesesuaian konfigurasi worker dengan kebutuhan pemrosesan.

$F_{resource}$ = Nilai efisiensi penggunaan sumber daya sistem.

F_{retry} = Nilai evaluasi berdasarkan tingkat *retry*.

$w_1 + w_2 + w_3 + w_4$ = Bobot masing-masing komponen evaluasi.

Komponen Throughput digunakan untuk mengevaluasi kemampuan sistem dalam menyelesaikan proses pengolahan gambar pada setiap tahapan *pipeline*. Semakin tinggi jumlah tugas yang dapat diproses dalam periode waktu tertentu, semakin tinggi pula nilai evaluasi yang diperoleh.

Komponen Parameter Match digunakan untuk mengevaluasi kesesuaian antara kapasitas *worker* dengan beban kerja pada setiap tahapan *pipeline*. Penilaian dilakukan dengan membandingkan jumlah *worker* yang tersedia terhadap kebutuhan pemrosesan yang ditunjukkan oleh panjang antrean (*queue*) dan jumlah *active task*. Konfigurasi *worker* yang terlalu sedikit dapat menyebabkan *bottleneck*, sedangkan konfigurasi yang terlalu besar mengakibatkan penggunaan sumber daya yang kurang efisien.

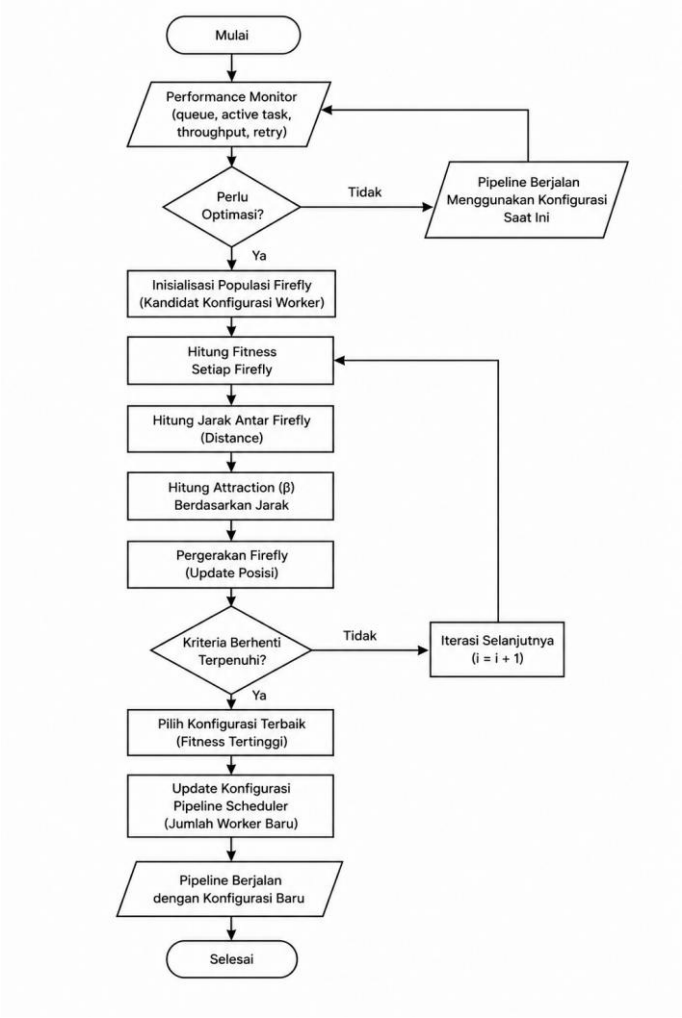
Komponen Resource Efficiency digunakan untuk mengevaluasi efisiensi penggunaan sumber daya selama proses pengolahan gambar berlangsung. Penilaian dilakukan berdasarkan kesesuaian jumlah *worker* terhadap beban kerja pada setiap tahapan *pipeline* serta mempertimbangkan kondisi penggunaan memori sistem agar proses optimasi tidak menghasilkan konfigurasi yang membebani sumber daya secara berlebihan.

Komponen Retry Rate digunakan untuk mengevaluasi tingkat keberhasilan proses berdasarkan jumlah tugas yang harus diproses ulang (*retry*). Semakin sedikit proses *retry* yang terjadi, semakin tinggi nilai evaluasi yang diperoleh sehingga konfigurasi tersebut dinilai lebih stabil.

Berdasarkan keempat komponen tersebut, setiap kandidat solusi memperoleh nilai *fitness* yang berbeda. Konfigurasi dengan nilai *fitness* tertinggi dipilih sebagai konfigurasi terbaik untuk memperbarui jumlah *Download Worker, OCR Worker, Translate Worker,* dan *Image Text Swapping Worker* pada *Pipeline Scheduler*.

3.6.5.2 Alur Optimasi Firefly Algorithm

Alur optimasi menggunakan Firefly Algorithm ditunjukkan pada Gambar 3.23.



Gambar 3.23 Alur Optimasi Firefly Algorithm

Berdasarkan Gambar 3.23, proses optimasi dimulai ketika Performance Monitor melakukan pemantauan terhadap kondisi *Pipeline Scheduler*. Informasi yang dipantau meliputi panjang antrean (*queue*), jumlah *active task*, nilai *throughput*, dan jumlah *retry* pada setiap tahapan *pipeline*. Informasi tersebut digunakan sebagai dasar untuk menentukan apakah konfigurasi *Pipeline Scheduler* masih mampu menangani beban kerja yang sedang diproses.

Apabila kondisi *pipeline* masih berada dalam batas yang telah ditentukan, proses pengolahan gambar akan terus berjalan menggunakan konfigurasi *worker* yang sedang digunakan. Sebaliknya, apabila terdeteksi penurunan performa, sistem akan memulai proses optimasi menggunakan Firefly Algorithm.

Tahap pertama dalam proses optimasi adalah membangkitkan sejumlah kandidat solusi (*firefly*). Setiap *firefly* merepresentasikan kombinasi jumlah *Download Worker, OCR Worker, Translate Worker,* dan *Image Text Swapping Worker* yang akan digunakan oleh *Pipeline Scheduler*. Selanjutnya setiap kandidat solusi dievaluasi menggunakan fungsi *fitness* untuk mengetahui kualitas konfigurasi yang dihasilkan.

Setelah seluruh kandidat solusi memperoleh nilai *fitness*, Firefly Algorithm melakukan proses pembaruan posisi (*position update*) berdasarkan mekanisme perpindahan antar-*firefly*. Proses ini dilakukan secara berulang sehingga setiap kandidat solusi bergerak menuju konfigurasi yang memiliki nilai *fitness* lebih baik. Pada setiap iterasi, nilai *fitness* dihitung kembali untuk mengevaluasi kualitas konfigurasi terbaru.

Proses optimasi berlangsung hingga memenuhi kriteria penghentian (*stopping criteria*), yaitu jumlah iterasi maksimum telah tercapai atau tidak ditemukan peningkatan nilai *fitness* yang signifikan. Setelah proses optimasi selesai, konfigurasi dengan nilai *fitness* terbaik dipilih sebagai konfigurasi baru *Pipeline Scheduler*.

Konfigurasi tersebut kemudian digunakan untuk memperbarui jumlah *Download Worker, OCR Worker, Translate Worker,* dan *Image Text Swapping Worker* sehingga proses pengolahan gambar selanjutnya dapat menyesuaikan dengan kondisi beban kerja yang sedang berlangsung. Dengan mekanisme tersebut, sistem diharapkan mampu menjaga kinerja *pipeline* secara adaptif tanpa perlu melakukan konfigurasi secara manual.

Tabel 3.10 Spesifikasi Modul Firefly Algorithm

Komponen	Deskripsi
Tujuan	Mengoptimalkan konfigurasi Pipeline Scheduler menggunakan Firefly Algorithm.
Input	*Queue*, *active task*, *throughput*, *retry*.
Proses	Deteksi bottleneck → Inisialisasi populasi → Perhitungan fitness → Pergerakan firefly → Seleksi konfigurasi terbaik → Pembaruan Pipeline Scheduler.
Output	Konfigurasi Pipeline Scheduler optimal berupa jumlah worker pada setiap tahapan pipeline.

3.7 Jadwal Penelitian

Penelitian ini direncanakan akan dilaksanakan selama 10 bulan, dimulai dari September 2025 hingga Juni 2026. Adapun jadwal kegiatan penelitian dapat dilihat pada berikut:

Tabel 3.11 Jadwal Penelitian

Tahapan Penelitian	Bulan 1		Bulan 2		Bulan 3		Bulan 4		Bulan 5		Bulan 6		Bulan 7		Bulan 8		Bulan 9		Bulan 10	
Minggu ke-	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4	1 & 2	3 & 4
Identifikasi Masalah																				
Studi Literatur																				
Analisis Kebutuhan Sistem																				
Perancangan Sistem																				
Implementasi Sistem																				
Pengujian dan Validasi																				
Analisis hasil dan penyusunan laporan akhir																				